KERJA PRAKTIK - KI141330

Pembuatan Modul Portofolio dan SKEM Aplikasi myITS Student-Connect

DIREKTORAT PENGEMBANGAN TEKNOLOGI DAN SISTEM INFORMASI INSTITUT TEKNOLOGI SEPULUH NOPEMBER

Periode: 6 Januari 2020 - 1 Desember 2020

Oleh:

Bintang Nuralamsyah	05111740000002
Ersad Ahmad Ishlahuddin	05111740000016

Pembimbing Jurusan
Hadziq Fabroyir, S.Kom., Ph.D.
Pembimbing Lapangan
Rizky Januar Akbar, S.Kom., M.Eng.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

KERJA PRAKTIK – KI141330

Pembuatan Modul Portofolio dan SKEM Aplikasi myITS StudentConnect

DIREKTORAT PENGEMBANGAN TEKNOLOGI DAN SISTEM INFORMASI INSTITUT TEKNOLOGI SEPULUH NOPEMBER Periode: 6 Januari 2020 - 1 Desember 2020

Oleh:
Bintang Nuralamsyah 05111740000002
Ersad Ahmad Ishlahuddin 05111740000016

Pembimbing Jurusan
Hadziq Fabroyir, S.Kom., Ph.D.
Pembimbing Lapangan
Rizky Januar Akbar, S.Kom., M.Eng.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

KERJA PRAKTIK

Pembuatan Modul Portofolio dan SKEM Aplikasi myITS StudentConnect

Oleh:

Bintang Nuralamsyah	05111740000002
Ersad Ahmad Ishlahuddin	05111740000016

Mengetahui,	Menyetujui,
DPTSI ITS	Dosen Pembimbing
Pembimbing Kerja Praktik	Kerja Praktik
Rizky Januar Akbar	Hadziq Fabroyir
NIP. 198701032014041001	NIP. 198602272019031006

[Halaman ini sengaja dikosongkan]

Pembuatan Modul Portofolio dan SKEM
Aplikasi myITS StudentConnect

Nama Mahasiswa	**: Bintang Nuralamsyah**
NRP	**: 05111740000002**
Nama Mahasiswa	**: Ersad Ahmad Ishlahuddin**
NRP	**: 05111740000016**
Departemen	**: Informatika FTIK-ITS**
Pembimbing Jurusan	**: Hadziq Fabroyir**
Pembimbing Lapangan	**: Rizky Januar Akbar**

ABSTRAK

Direktorat Pengembangan Teknologi dan Sistem Informasi (DPTSI) bertugas untuk menyediakan dan mengelola layanan Teknologi Informasi di lingkungan ITS. DPTSI berperan mendukung aktivitas akademik, penelitian dan pengabdian masyarakat, serta manajerial di lingkungan ITS guna membantu ITS mencapai visi misinya. ITS memiliki beberapa modul yang berkaitan dengan proses bisnis di Direktorat Kemahasiswaan. Diantara modul tersebut ialah modul beasiswa, modul SKEM, modul SKPI, modul ormawa, modul simbelmawa, dan modul prestasi. Masing - masing sistem melakukan input data secara mandiri sehingga memungkinkan adanya duplikasi data. Oleh karena itu, myITS Student Connect dirancang sebagai sistem informasi satu pintu berbasi portofolio untuk seluruh proses bisnis atau modul–modul kemahasiswaan di atas agar mahasiswa dapat memasukkan datanya dan dapat memanfaatkan datanya untuk berbagai kepentingan.

Modul Portofolio dan SKEM Aplikasi myITS StudentConnect ini dibuat dengan menggunakan bahasa pemrograman web seperti PHP, HTML, CSS dan Javascript dengan menggunakan DBMS Microsoft SQL Server. Dengan aplikasi ini nantinya diharapkan mahasiswa ITS dapat memiliki satu aplikasi yang datanya dapat digunakan untuk banyak keperluan, sedangkan dari pihak ITS dapat memanfaatkan data kemahasiswaan dengan lebih baik

Kata kunci: Sistem Informasi, Aplikasi Web

KATA PENGANTAR

Puji syukur kami haturkan kepada Allah SWT karena berkat rahmat-Nya kami dapat melaksanakan salah satu kewajiban kami sebagai mahasiswa Departemen Informatika, yakni Kerja Praktik(KP).

Kami menyadari masih ada kekurangan baik dalam pelaksanaan Kerja Praktik maupun penyusunan buku laporan ini. Namun, kami berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Kami mengharapkan kritik dan saran yang membangun untuk kesempurnaan buku laporan Kerja Praktikini.

Melalui buku ini, kami juga ingin menyampaikan rasa terima kasih kepada orang-orang yang telah membantu, baik langsung maupun tidak langsung, dalam pelaksanaan Kerja Praktik hingga penyusunan laporan. Orang-orang tersebut antara lain adalah:

1. Kedua orang tua penulis.
2. Bapak Rizky Januar Akbar, S.Kom., M.Eng., selaku pembimbing lapangan kami di Direktorat Pengembangan Teknologi Dan Sistem Informasi (DPTSI) ITS
3. Bapak Hadziq Fabroyir, S.Kom., Ph.D., selaku dosen pembimbing Kerja Praktik.
4. Bapak Ary Mazharuddin Shiddiqi S.Kom., M.Comp., Ph.D., selaku koordinator Kerja Praktik.

Surabaya, Januari 2021

Penulis

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1. Latar Belakang

Demi memberikan layanan yang terbaik untuk mahasiswa, Institut Teknologi Sepuluh Nopember memiliki beberapa modul yang berkaitan dengan proses bisnis di Direktorat Kemahasiswaan. Beberapa modul tersebut ialah modul beasiswa, modul SKEM, modul SKPI, modul ormawa, modul simbelmawa, dan modul prestasi. Masing-masing system melakukan input data secara mandiri sehingga memungkinkan adanya duplikasi data.

Hal tersebut menyebabkan data yang telah diinputkan tidak dapat dimanfaatkan dengan baik untuk kepentingan lain seperti akreditasi prodi atau PT, mencari kandidat mahasiswa berprestasi, dan pelaporan monev. Salah satu alternatif solusi untuk menangani permasalahan tersebut adalah dengan membuat suatu aplikasi yang bersifat single input multi purpose, dimana data cukup diinputkan sekali dan data tersebut dapat dimanfaatkan untuk berbagai keperluan.

1.2. Tujuan

Tujuan Kerja Praktik ini adalah untuk menyelesaikan kewajiban kuliah Kerja Praktik di Institut Teknologi Sepuluh Nopember dengan beban dua SKS. Selain itu, Kerja Praktik ini juga bertujuan untuk mengembangkan sistem informasi satu pintu untuk kemahasiswaan sebagai media agar mahasiswa dapat memasukkan datanya dan dapat memanfaatkan datanya untuk berbagai kepentingan.

1.3. Manfaat

Berikut adalah manfaat yang diperoleh dari Kerja Praktik pembuatan Modul Portofolio dan SKEM Aplikasi myITS StudentConnect:

- Dapat memudahkan mahasiswa dalam memasukkan data-data portofolio dan menggunakannya kembali.
- Dapat memudahkan ITS dalam memanfaatkan data kemahasiswaan dengan lebih baik.

1.4. Rumusan Permasalahan

Berikut ini rumusan masalah pada Kerja Praktik pembuatan Modul Portfolio dan SKEM Aplikasi myITS StudentConnect:

- Bagaimana cara mahasiswa dapat menginputkan data non akademik dan memanfaatkannya kembali?
- Bagaimana cara Institut Teknologi Sepuluh Nopember memverifikasi kebenaran data yang diinputkan mahasiswa?
- Bagaimana cara mahasiswa menggunakan data portofolio yang telah diinputkan untuk keperluan SKEM?

1.5. Lokasi dan Waktu Kerja Praktik

Kerja Praktik ini dilaksanakan pada waktu dan tempat sebagai berikut:

Lokasi	: Direktorat Pengembangan Teknologi dan Sistem Informasi Institut Teknologi Sepuluh Nopember
Alamat	: Jl. Teknik Kimia, Keputih, Sukolilo, Surabaya
Waktu	: 6 Januari 2020 - 1 Desember 2020
Hari Kerja	: Senin - Jumat
Jam Kerja	: 08.00 WIB - 16.00 WIB

1.6. Metodologi Kerja Praktik

1. **Perumusan Masalah**

Untuk mengetahui domain dan fungsionalitas, dijelaskan secara rinci bagaimana sistem yang harus dibuat. Penjelasan oleh pembimbing lapangan Kerja Praktik kali ini menghasilkan beberapa catatan mengenai gambaran secara garis besar tentang kebutuhan atau fitur apa saja yang harus ada di dalam website. Setelah mendapatkan gambaran sistem, diskusi lebih lanjut dilakukan guna menentukan rancangan serta *tools* pendukung pembuatan sistem.

2. Studi Literatur

Pada tahap ini, setelah ditentukannya rancangan *database*, bahasa pemrograman sampai dengan teknologi beserta *tools* tambahan yang digunakan, dilakukan studi literatur lanjut mengenai bagaimana penggunaannya dalam membangun sistem sesuai yang diharapkan.

Aplikasi yang akan dibuat merupakan system yang akan dibangun, ada beberapa tools yang digunakan. Untuk frontend digunakan Bahasa pemrograman HTML, CSS dan JavaScript. Sedangkan untuk backend digunakan Bahasa pemrograman PHP. Ada beberapa tools tambahan yang mendukung pembuatan website tersebut yaitu XAMPP dan Visual Studio Code. Framework yang digunakan yaitu Laravel. Database yang digunakan yaitu SQLServer.

3. Analisis dan Perancangan Sistem

Langkah ini meliputi penjelasan awal tentang sistem. Bagaimana cara kerja sistem dengan skenario tertentu. Dari penjelasan awal telah didapatkan beberapa kebutuhan fungsional secara garis besar. Kemudian dilanjutkan dengan memperjelas dan menspesifikkan kebutuhan-kebutuhan tersebut. Dilanjutkan berdiskusi dengan pembimbing lapangan untuk mengetahui apakah

kebutuhan-kebutuhan tersebut sudah tepat.

4. **Implementasi Sistem**

   Implementasi sistem didasarkan oleh perancangan dan analisis sebelumnya. Penentuan atribut atau fitur yang akan digunakan pada model juga didasari pada analisis sebelumnya. Penentuan tipe data dan format keluaran juga disesuaikan dengan kebutuhan.

5. **Pengujian dan Evaluasi**

   Pengujian dilakukan oleh pembimbing lapangan dan anggota tim lain setiap fitur yang sudah selesai untuk memberikan evaluasi ketika ada yang tidak sesuai, dan persetujuan apabila sudah sesuai.

1.7. Sistematika Laporan

Laporan Kerja Praktik ini terdiri dari 7 bab dengan rincian sebagai berikut:

1. Bab I: Pendahuluan

Pada bab ini dijelaskan tentang latar belakang permasalahan, tujuan, waktu pelaksanaan, serta sistematika pengerjaan Kerja Praktik dan juga penulisan laporan Kerja Praktik.

2. Bab II: Profil Instansi

Pada bab ini, dijelaskan secara rinci tentang profil perusahaan tempat kami melaksanakan Kerja Praktik, yakni Direktorat Pengembangan Teknologi Dan Sistem Informasi Institut Teknologi Sepuluh Nopember.

3. Bab III: Tinjauan Pustaka

Pada bab ini, dijelaskan mengenai tinjauan pustaka dan literatur yang digunakan dalam penyelesaian Kerja Praktik di Direktorat Pengembangan Teknologi dan Sistem Informasi Institut Teknologi Sepuluh Nopember.

4. Bab IV: Analisis dan Perancangan Sistem

Dalam bab ini dibahas tentang proses analisis kebutuhan berdasarkan kondisi di lapangan dan perancangannya yang meliputi desain aplikasi yang akan dikembangkan. Proses analisis dan desain aplikasi menghasilkan daftar fitur yang dibutuhkan.

5. Bab V: Desain Model dan Implementasi Sistem

Dalam bab ini dibahas tentang desain model dan implementasi secara keseluruhan.

6. Bab VI: Pengujian dan Evaluasi

Dalam bab ini dibahas tentang skenario pengujian, dan evaluasi pengujian setelah model selesai dibangun.

7. Bab VII: Kesimpulan dan Saran

Bab ini berisi tentang kesimpulan dan saran yang didapatkan dari tugas selama Kerja Praktik.

[Halaman ini sengaja dikosongkan]

BAB II PROFIL INSTANSI

2.1. Profil DPTSI ITS

Direktorat Pengembangan Teknologi dan Sistem Informasi (DPTSI) bertugas untuk menyediakan dan mengelola layanan Teknologi Informasi di lingkungan ITS. Terkait peran, DPTSI berperan untuk mendukung aktivitas akademik, penelitian dan pengabdian masyarakat, serta manajerial di lingkungan ITS dalam rangka membantu ITS mencapai visi misinya.

2.2. Visi dan Misi DPTSI ITS

Visi dan misi dari Direktorat Pengembangan Teknologi dan Sistem Informasi Institut Teknologi Sepuluh Nopember adalah sebagai berikut:

- Visi: Membangun Budaya dan Transformasi Digital untuk Peningkatan Kualitas Kinerja dan Layanan Berbasis TIK dalam rangka mewujudkan ITS sebagai “World Class Research and Innovative University”
- Misi:
    - Peningkatan Kapasitas Jaringan
    - Pengembangan & Pemeliharaan Sistem Informasi Terintegrasi menuju ITS in One Hand
    - Peningkatan Efisiensi dan Efektifitas Pelaporan dengan ITS Satu Data
    - Peningkatan Layanan Prima berbasis Digital

[Halaman ini sengaja dikosongkan]

BAB III
TINJAUAN PUSTAKA

3.1. Pemrogram Web

Pemrograman merupakan sekumpulan intruksi atau perintah tertulis yang di buat oleh manusia secara logis untuk memerintahkan komputer agar melakukan langkah atau proses tertentu dalam menyelesaikan suatu masalah. Web sendiri merupakan sebuah halaman atau media informasi yang dapat diakses dengan perangkat lunak browser melalui jaringan komputer atau internet. Pemrograman web adalah proses membuat aplikasi komputer yang dapat digunakan atau ditampilkan dengan bantuan browser.

3.2. HTML

Hyper Text Markup Language (HTML) adalah sebuah *markup languange* yang digunakan untuk membuat sebuah halaman web, menampilkan berbagai informasi di dalam sebuah penjelajah web Internet dan pemformatan hiperteks sederhana yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegerasi.

3.3. CSS

Cascading Style Sheets (CSS) merupakan mekanisme sederhana untuk menambahkan style (seperti warna tulisan, font, jarak tulisan) ke dalam dokumen web. Hampir semua browser dan banyak aplikasi yang ada saat ini telah menunjang penggunaan CSS.

3.4. Javascript

JavaScript adalah sebuah Bahasa pemrograman yang memungkinkan kita mengimplementasikan hal-hal kompleks pada halaman web terutama yang bersifat

interaktif. Javascript juga dapat digunakan untuk memanipulasi dan mengirim data pada browser pengguna.

3.5. PHP

PHP pada dasarnya merupakan singkatan dari PHP: Hypertext Prepocessor. PHP digunakan sebagai salah satu script untuk membuat fungsi-fungsi sebagai mempermudah sistem teknis dalam website. Dalam praktiknya PHP biasanya digunakan bersama dengan penggunaan bahasa pemrograman lainnya seperti bahasa pemrograman HTML dan bahasa pemrograman JavaScript.

3.6. Laravel

Laravel adalah sebuah framework PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (model view controller). Laravel adalah pengembangan website berbasis MVP yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan sintaks yang ekspresif, jelas dan menghemat waktu.

3.7. Visual Studio Code

Visual Studio Code adalah editor kode yang dikembangkan oleh Microsoft untuk Windows, Linux dan macOS. Visual Studio Code memberikan dukungan untuk debugging, kontrol Git tertanam, penyorotan sintaksis, penyelesaian kode cerdas, snippet, dan refactoring kode. Visual Studio Code juga dapat disesuaikan, sehingga pengguna dapat mengubah tema editor, pintasan keyboard, dan preferensi.

3.8. Microsoft SQL Server

Microsoft SQL Server adalah sebuah sistem manajemen basis data relasional (RDBMS) produk Microsoft. Bahasa kueri utamanya adalah Transact-SQL yang merupakan implementasi dari SQL standar ANSI/ISO yang digunakan oleh Microsoft dan Sybase. Umumnya SQL Server digunakan di dunia bisnis yang memiliki basis data berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya SQL Server pada basis data besar.

[Halaman ini sengaja dikosongkan]

BAB IV
ANALISIS DAN DESAIN

4.1. Analisis Sistem

4.1.1. Definisi Umum Fitur

Terdapat 2 *stakeholder* utama dalam modul portofolio dan SKEM Aplikasi myITS StudentConnect, yaitu mahasiswa dan verifikator. Mahasiswa dapat memasukkan maupun menyunting berbagai data terkait aktivitas nonakademik pada modul portofolio dan mengajukan SKEM pada modul SKEM. Sedangkan verifikator dapat memverifikasi data-data yang diinputkan pada modul portofolio dan ajuan SKEM mahasiswa pada modul SKEM.

4.1.2. Analisis Kebutuhan Fungsional

Beberapa kebutuhan fungsional yang dibutuhkan pada modul portofolio dan SKEM aplikasi myITS StudentConnect dapat dilihat pada tabel 1.

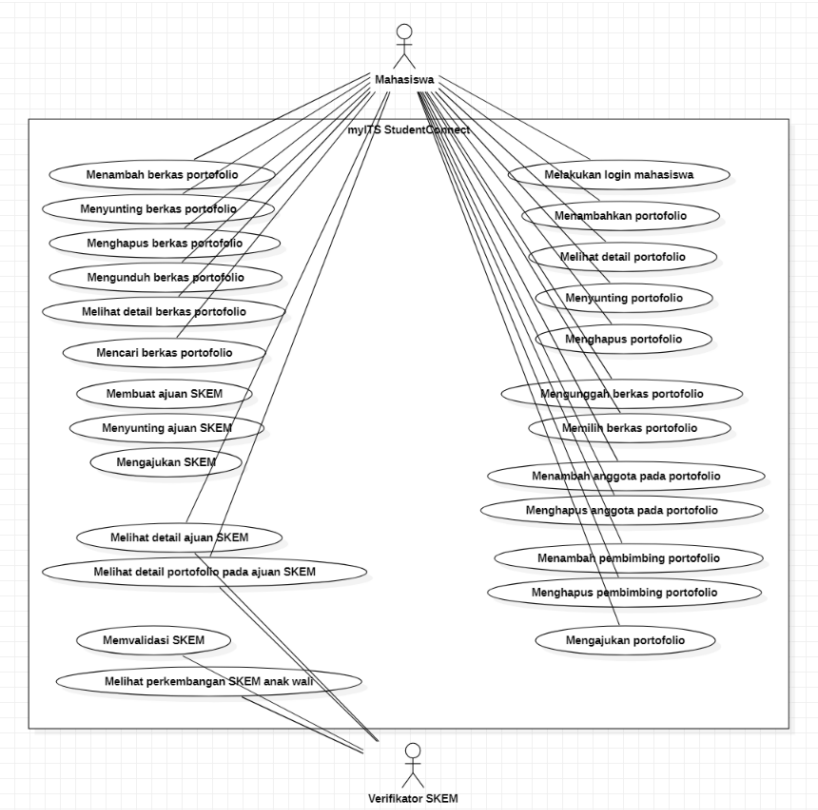
Tabel 1: Kebutuhan fungsional

Kode Kebutuhan	Deskripsi Kebutuhan
FR-001	Melakukan login mahasiswa
FR-002	Menambahkan portofolio
FR-003	Melihat detail portofolio
FR-004	Menyunting portofolio
FR-005	Menghapus portofolio
FR-006	Mengunggah berkas portofolio
FR-007	Memilih berkas portofolio
FR-008	Menambah anggota pada portofolio
FR-009	Menghapus anggota pada portofolio

FR-010	Menambah pembimbing portofolio
FR-011	Menghapus pembimbing portofolio
FR-012	Mengajukan portofolio
FR-013	Menambah berkas portofolio
FR-014	Menyunting berkas portofolio
FR-015	Menghapus berkas portofolio
FR-016	Mengunduh berkas portofolio
FR-017	Melihat detail berkas portofolio
FR-018	Mencari berkas portofolio
FR-019	Membuat ajuan SKEM
FR-020	Menyunting ajuan SKEM
FR-021	Mengajukan SKEM
FR-022	Melihat detail ajuan SKEM
FR-023	Melihat detail portofolio pada ajuan SKEM
FR-024	Memvalidasi SKEM
FR-025	Melihat perkembangan SKEM anak wali

4.2. Diagram Kasus Penggunaan

Pembahasan dengan pembimbing lapangan tentang fitur-fitur yang perlu ada dalam modul portofolio dan modul SKEM Aplikasi myITS StudentConnect menghasilkan beberapa fitur yang dijadikan diagram kasus penggunaan (Use Case Diagram) sehingga memudahkan untuk dipahami. Use Case Diagram yang telah dibuat dapat dilaht pada Gambar 1.



Gambar 1: Diagram Use Case

4.3. Spesifikasi Kasus Penggunaan

4.3.1. Melakukan login mahasiswa

Tabel 2: Spesifikasi Use Case melakukan login mahasiswa

Kode Use Case	UC001
Nama Use Case	Melakukan login mahasiswa
Aktor	Mahasiswa

Deskripsi	Proses masuk mahasiswa ke dalam sistem
Kondisi Awal	Mahasiswa belum masuk ke dalam sistem
Kondisi Akhir	Mahasiswa telah masuk ke dalam sistem
Alur Normal	
Aktor	Sistem
1. Mahasiswa membuka halaman login untuk admin 3. Mahasiswa memasukkan username dan password myITS SSO 4. Mahasiswa menekan tombol Sign in	2. Sistem menampilkan form untuk login 5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak sesuai 6. Sistem membuka halaman beranda mahasiswa
Alur Alternatif	
A.1. Data yang dimasukkan tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang dimasukkan tidak sesuai

4.3.2. Menambahkan portofolio

Tabel 3: Spesifikasi Use Case menambahkan portofolio

Kode Use Case	UC002
Nama Use Case	Menambahkan portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menambahkan portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem dan portofolio belum ditambahkan
Kondisi Akhir	Portofolio telah ditambahkan dengan status ‘Belum diajukan’
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Tambah Ajuan’ 3. Mahasiswa mengisi form ajuan portofolio 4. Mahasiswa menekan tombol simpan E.1. Mahasiswa menekan tombol ‘Kembali’	2. Sistem menampilkan halaman form tambah ajuan portofolio 5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap

	6. Sistem menyimpan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem menampilkan halaman daftar portofolio

4.3.3. Melihat detail portofolio

Tabel 4: Spesifikasi Use Case melihat detail portofolio

Kode Use Case	UC003
Nama Use Case	Melihat detail portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk melihat detail portofolio mahasiswa yang telah ditambahkan
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem
Kondisi Akhir	Mahasiswa dapat melihat detail portofolio
Alur Normal	
Aktor	Sistem

1. Mahasiswa menekan salah satu portofolio pada daftar portofolio	2. Sistem menampilkan halaman detail portofolio

4.3.4. Menyunting portofolio

Tabel 5: Spesifikasi Use Case menyunting portofolio

Kode Use Case	UC004
Nama Use Case	Menyunting portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menyunting portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Portofolio telah disunting
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’	2. Sistem menampilkan halaman form sunting ajuan portofolio
3. Mahasiswa menyunting isian form ajuan portofolio	

4. Mahasiswa menekan tombol simpan E.1. Mahasiswa menekan tombol 'Kembali'	5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap 6. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem menampilkan halaman detail portofolio

4.3.5. Menghapus portofolio

Tabel 6: Spesifikasi Use Case menghapus portofolio

Kode Use Case	UC004
Nama Use Case	Menghapus portofolio

Aktor	Mahasiswa
Deskripsi	Proses untuk menghapus portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio belum dihapus
Kondisi Akhir	Portofolio telah dihapus
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol hapus portofolio 3. Mahasiswa menekan tombol hapus E.1. Mahasiswa menekan tombol batal	2. Sistem menampilkan modal konfirmasi hapus portofolio 4. Sistem menghapus data dari database
Alur Eksepsi	
E.1. Mahasiswa menekan tombol batal	
Aktor	Sistem
	1. Sistem menghilangkan modal konfirmasi hapus portofolio

4.3.6. Mengunggah berkas portofolio

Tabel 7: Spesifikasi Use Case mengunggah berkas portofolio

Kode Use Case	UC006
Nama Use Case	Mengunggah berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk mengunggah berkas portofolio
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Berkas portofolio telah diunggah
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Kelengakapan berkas’	
	2. Sistem menampilkan halaman form sunting berkas portofolio
3. Mahasiswa menekan tombol ‘Lampirkan berkas’	
4. Mahasiswa menekan tombol ‘Unggah berkas’	
	5. Sistem menampilkan modal untuk unggah berkas
6. Mahasiswa mengisi form unggah berkas	
7. Mahasiswa menekan tombol ‘Kirim’	

E.1. Mahasiswa menekan tanda silang	5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai 6. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tanda silang	
Aktor	Sistem
	1. Sistem menghilangkan modal untuk unggah berkas

4.3.7. Memilih berkas portofolio

Tabel 8: Spesifikasi Use Case memilih berkas portofolio

Kode Use Case	UC007
Nama Use Case	Memilih berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk memilih berkas portofolio

Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Berkas portofolio telah dipilih
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Kelengakapan berkas’	
	2. Sistem menampilkan halaman form sunting berkas portofolio
3. Mahasiswa menekan tombol ‘Lampirkan berkas’ 4. Mahasiswa menekan tombol ‘Pilih dari berkas saya’	
	5. Sistem menampilkan modal untuk pilih berkas
6. Mahasiswa memilih berkas dengan menekan checkbox 7. Mahasiswa menekan tombol ‘Kirim’ E.1. Mahasiswa menekan tanda silang	
	5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai

	6. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tanda silang	
Aktor	Sistem
	1. Sistem menghilangkan modal untuk unggah berkas

4.3.8. Menambah anggota pada portofolio

Tabel 9: Spesifikasi Use Case menambah anggota pada portofolio

Kode Use Case	UC008
Nama Use Case	Menambah anggota pada portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menambah anggota pada portofolio
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’

Kondisi Akhir	Anggota pada portofolio telah ditambahkan
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Anggota’	
	2. Sistem menampilkan halaman form sunting anggota pada portofolio
3. Mahasiswa menekan tombol ‘Tambah Anggota lain’	
4. Mahasiswa mencari anggota dengan nama atau NRP	
5. Mahasiswa menekan tombol ‘Simpan’	
E.1. Mahasiswa menekan tombol ‘Kembali’	6. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai 7. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
2. Mahasiswa membaca pesan	

tersebut 3. Mahasiswa kembali ke alur normal nomor 3	
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem menampilkan halaman detail portofolio

4.3.9. Menghapus anggota pada portofolio

Tabel 10: Spesifikasi Use Case menghapus anggota pada portofolio

Kode Use Case	UC009
Nama Use Case	Menghapus anggota pada portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menghapus anggota pada portofolio
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Anggota pada portofolio telah dihapus
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Anggota’	

3. Mahasiswa menekan tanda titik tiga di kanan anggota 4. Mahasiswa menekan tombol ‘Hapus’ 6. Mahasiswa menekan tombol ‘Hapus’ E.1. Mahasiswa menekan tombol ‘Batal’	2. Sistem menampilkan halaman form sunting anggota pada portofolio 5. Sistem menampilkan modal konfirmasi hapus anggota 7. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai 8. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi

ke alur normal nomor 3	
Alur Eksepsi	
E.1. Mahasiswa menekan tombol batal	
Aktor	Sistem
	1. Sistem menghilangkan modal konfirmasi hapus anggota

4.3.10. Menambah pembimbing portofolio

Tabel 11: Spesifikasi Use Case menambah pembimbing portofolio

Kode Use Case	UC010
Nama Use Case	Menambah pembimbing portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menambah pembimbing pada portofolio
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Pembimbing pada portofolio telah ditambahkan
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Pembimbing’	2. Sistem menampilkan halaman form sunting pembimbing pada portofolio

3. Mahasiswa menekan tombol ‘Tambah Pembimbing’lain’ 4. Mahasiswa mencari pembimbing dengan nama atau NIP 5. Mahasiswa menekan tombol ‘Simpan’ E.1. Mahasiswa menekan tombol ‘Kembali’	6. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai 7. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem menampilkan halaman detail portofolio

4.3.11. Menghapus pembimbing portofolio

Tabel 12: Spesifikasi Use Case menghapus pembimbing portofolio

Kode Use Case	UC011
Nama Use Case	Menghapus pembimbing pada portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menghapus pembimbing pada portofolio
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan portofolio dengan status ‘Belum diajukan’ atau ‘Revisi’
Kondisi Akhir	Pembimbing pada portofolio telah dihapus
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Sunting’ pada bagian ‘Pembimbing’	
	2. Sistem menampilkan halaman form sunting pembimbing pada portofolio
3. Mahasiswa menekan tanda titik tiga di kanan pembimbing	
4. Mahasiswa menekan tombol ‘Hapus’	
	5. Sistem menampilkan modal konfirmasi hapus pembimbing

6. Mahasiswa menekan tombol ‘Hapus’ E.1. Mahasiswa menekan tombol ‘Batal’	7. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai 8. Sistem menyimpan perubahan data portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap atau tidak sesuai	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 3	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tombol batal	
Aktor	Sistem
	1. Sistem menghilangkan modal konfirmasi hapus pembimbing

4.3.12. Mengajukan portofolio

Tabel 13: Spesifikasi Use Case mengajukan portofolio

Kode Use Case	UC012
Nama Use Case	Mengajukan portofolio

Aktor	Mahasiswa
Deskripsi	Proses untuk mengajukan portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka halaman detail portofolio dan kelengkapan berkas portofolio telah terpenuhi
Kondisi Akhir	Portofolio telah diajukan
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol 'Ajukan'	2. Sistem menyimpan perubahan data portofolio ke dalam database

4.3.13. Menambah berkas portofolio

Tabel 14: Spesifikasi Use Case menambah berkas portofolio

Kode Use Case	UC013
Nama Use Case	Menambahkan berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menambahkan berkas portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya dan berkas belum ditambahkan
Kondisi Akhir	Portofolio telah ditambahkan
Alur Normal	
Aktor	Sistem

1. Mahasiswa menekan tombol ‘Unggah Berkas’	
	2. Sistem menampilkan modal form tambah berkas portofolio
3. Mahasiswa mengisi form berkas portofolio	
4. Mahasiswa menekan tombol kirim	
E.1. Mahasiswa menekan tanda silang	5. Sistem memeriksa data yang dimasukkan
	A.1. Data yang dimasukkan tidak lengkap
	6. Sistem menyimpan data berkas portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap	
Aktor	Sistem
	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
2. Mahasiswa membaca pesan tersebut	
3. Mahasiswa kembali ke alur normal nomor 1	
Alur Eksepsi	
E.1. Mahasiswa menekan tanda silang	
Aktor	Sistem

	1. Sistem menghilangkan modal form tambah berkas portofolio

4.3.14. Menyunting berkas portofolio

Tabel 15: Spesifikasi Use Case menyunting berkas portofolio

Kode Use Case	UC014
Nama Use Case	Menyunting berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menyunting berkas portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya dan berkas belum disunting
Kondisi Akhir	Portofolio telah disunting
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tanda titik tiga pada berkas 2. Mahasiswa menekan tombol sunting 4. Mahasiswa mengisi form sunting berkas portofolio 5. Mahasiswa menekan tombol kirim E.1. Mahasiswa menekan tanda silang	3. Sistem menampilkan modal form sunting berkas portofolio

	6. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap 7. Sistem menyimpan data berkas portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap	
Aktor	Sistem
2. Mahasiswa membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 1	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi
Alur Eksepsi	
E.1. Mahasiswa menekan tanda silang	
Aktor	Sistem
	1. Sistem menghilangkan modal form sunting berkas portofolio

4.3.15. Menghapus berkas portofolio

Tabel 16: Spesifikasi Use Case menghapus berkas portofolio

Kode Use Case	UC015
Nama Use Case	Menghapus berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk menghapus berkas portofolio mahasiswa

Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya dan berkas belum dihapus
Kondisi Akhir	Portofolio telah dihapus
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tanda titik tiga pada berkas 2. Mahasiswa menekan tombol hapus 4. Mahasiswa menekan tombol kirim E.1. Mahasiswa menekan tanda silang	3. Sistem menampilkan modal form konfirmasi hapus berkas portofolio 5. Sistem memeriksa data yang dimasukkan A.1. Data yang dimasukkan tidak lengkap 6. Sistem menyimpan data berkas portofolio ke dalam database
Alur Alternatif	
A.1. Data yang dimasukkan tidak lengkap	
Aktor	Sistem
2. Mahasiswa	1. Sistem menampilkan pesan error bahwa terdapat data yang belum lengkap/belum diisi

membaca pesan tersebut 3. Mahasiswa kembali ke alur normal nomor 1	
Alur Eksepsi	
E.1. Mahasiswa menekan tanda silang	
Aktor	Sistem
	1. Sistem menghilangkan modal form konfirmasi hapus berkas portofolio

4.3.16. Mengunduh berkas portofolio

Tabel 17: Spesifikasi Use Case mengunduh berkas portofolio

Kode Use Case	UC016
Nama Use Case	Mengunduh berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk mengunduh berkas portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya dan berkas belum diunduh
Kondisi Akhir	Portofolio telah diunduh
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tanda titik tiga pada berkas 2. Mahasiswa menekan tombol unduh	

	3. Sistem mengunduh berkas

4.3.17. Melihat detail berkas portofolio

Tabel 18: Spesifikasi Use Case melihat detail berkas portofolio

Kode Use Case	UC017
Nama Use Case	Melihat detail berkas portofolio
Aktor	Mahasiswa
Deskripsi	Proses untuk melihat detail berkas portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya
Kondisi Akhir	Mahasiswa telah melihat detail berkas portofolio
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tanda titik tiga pada berkas 2. Mahasiswa menekan tombol detail	3. Sistem menampilkan form detail berkas portofolio

4.3.18. Mencari berkas portofolio

Tabel 19: Spesifikasi Use Case mencari berkas portofolio

Kode Use Case	UC018
Nama Use Case	Mencari detail berkas portofolio
Aktor	Mahasiswa

Deskripsi	Proses untuk mencari berkas portofolio mahasiswa
Kondisi Awal	Mahasiswa telah melakukan masuk ke sistem, membuka menu berkas saya
Kondisi Akhir	Mahasiswa telah mencari berkas
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Atur’ 2. Mahasiswa mengisi filter sesuai kebutuhan 3. Mahasiswa menekan tanda cari	4. Sistem mencari berkas sesuai filter 5. Sistem menampilkan hasil pencarian

4.3.19. Membuat ajuan SKEM

Tabel 20: Spesifikasi Use Case membuat ajuan SKEM

Kode Use Case	UC019
Nama Use Case	Membuat ajuan SKEM
Aktor	Mahasiswa
Deskripsi	Proses bagi mahasiswa untuk membuat ajuan SKEM
Kondisi Awal	Mahasiswa telah masuk ke dalam sistem dan membuka menu SKEM
Kondisi Akhir	Ajuan SKEM terbuat
Alur Normal	
Aktor	Sistem

1. Mahasiswa menekan tombol buat pada semester yang dikehendaki 3. Mahasiswa memilih portofolio-portofolio yang ingin ditambahkan dan menekan tombol buat E.1. Mahasiswa menekan tombol kembali	2. Sistem menampilkan portofolio-portofolio yang dapat ditambahkan ke dalam ajuan 4. Sistem menambahkan ajuan SKEM dengan status “belum diajukan” pada database
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem mengembalikan pengguna pada menu awal SKEM

4.3.20. Menyunting ajuan SKEM

Tabel 21: Spesifikasi Use Case menyunting ajuan SKEM

Kode Use Case	UC020
Nama Use Case	Menyunting ajuan SKEM
Aktor	Mahasiswa
Deskripsi	Proses bagi mahasiswa untuk menyunting ajuan SKEM yang telah dibuat namun belum diajukan / divalidasi

Kondisi Awal	Mahasiswa telah masuk ke dalam sistem dan membuka menu SKEM
Kondisi Akhir	Ajuan SKEM tersunting
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Detail Ajuan’ pada semester yang dikehendaki	
	2. Sistem menampilkan detail ajuan SKEM yang dipilih
3. Mahasiswa menekan tombol sunting	
	4. Sistem menampilkan portofolio-portofolio yang dapat diajukan
5. Mahasiswa memilih portofolio-portofolio yang ingin diajukan dan menekan tombol simpan E.1 Mahasiswa menekan tombol kembali	
	6. Sistem menyimpan perubahan pada ajuan SKEM
Alur Eksepsi	
E.1. Mahasiswa menekan tombol kembali	
Aktor	Sistem
	1. Sistem mengembalikan pengguna pada laman detail ajuan SKEM yang dipilih

4.3.21. Mengajukan SKEM

Tabel 22: Spesifikasi Use Case mengajukan SKEM

Kode Use Case	UC021
Nama Use Case	Mengajukan SKEM

Aktor	Mahasiswa
Deskripsi	Proses bagi mahasiswa untuk mengajukan SKEM yang telah dibuat atau disunting
Kondisi Awal	Mahasiswa telah masuk ke dalam sistem dan membuka menu SKEM
Kondisi Akhir	Status ajuan SKEM menjadi ‘sedang diajukan’ dan siap diperiksa dosen wali
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Detail Ajuan’ pada semester yang dikehendaki	
	2. Sistem menampilkan detail ajuan SKEM yang dipilih
3. Mahasiswa menekan tombol ‘ajukan’ E.1 SKEM tidak memenuhi persyaratan untuk diajukan	
	4. Sistem menampilkan konfirmasi untuk mengajukan SKEM
5. Mahasiswa menekan tombol ‘ajukan’ E.2 Mahasiswa menekan tombol batal	
	6. Sistem merubah status ajuan SKEM menjadi ‘sedang diajukan’
Alur Eksepsi	

E.1. SKEM tidak memenuhi persyaratan untuk tidak diajukan	
Aktor	Sistem
	1. Sistem menampilkan laman detail SKEM tanpa ada tombol ‘ajukan’
E.2 Mahasiswa menekan tombol batal	
	1. Sistem mengembalikan pengguna ke laman detail ajuan SKEM yang dipilih

4.3.22. Melihat detail ajuan SKEM

Tabel 23: Spesifikasi Use Case melihat detail ajuan SKEM

Kode Use Case	UC022
Nama Use Case	Melihat detail ajuan SKEM
Aktor	Mahasiswa
Deskripsi	Proses bagi mahasiswa untuk melihat detail ajuan SKEM yang telah dibuat
Kondisi Awal	Mahasiswa telah masuk ke dalam sistem dan membuka menu SKEM
Kondisi Akhir	Detail ajuan SKEM ditampilkan
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Detail Ajuan’ pada semester yang dikehendaki E.1 Ajuan SKEM belum dibuat	2. Sistem menampilkan detail ajuan SKEM yang dipilih
Alur Eksepsi	
E.1. Ajuan SKEM belum dibuat	

Aktor	Sistem
	1. tombol ‘Detail Ajuan’ tidak akan muncul pada semester yang dikehendaki

4.3.23. Melihat detail portofolio pada ajuan SKEM

Tabel 24: Spesifikasi Use Case melihat detail portofolio pada ajuan SKEM

Kode Use Case	UC023
Nama Use Case	Melihat detail portofolio pada ajuan SKEM
Aktor	Mahasiswa
Deskripsi	Proses bagi mahasiswa untuk melihat detail hasil penilaian portofolio yang diajukan pada SKEM
Kondisi Awal	Mahasiswa telah masuk ke dalam sistem dan membuka menu SKEM
Kondisi Akhir	Detail penilaian portofolio ditampilkan
Alur Normal	
Aktor	Sistem
1. Mahasiswa menekan tombol ‘Detail Ajuan’ pada semester yang dikehendaki	
	2. Sistem menampilkan detail ajuan SKEM yang dipilih
3. Mahasiswa menekan salah satu nama portofolio pada laman detail ajuan SKEM	

E.1 Ajuan SKEM tidak memiliki portofolio	4. Sistem menampilkan laman berisi detail penilaian dari portofolio yang dipilih
Alur Eksepsi	
E.1. Ajuan SKEM tidak memiliki portofolio	
Aktor	Sistem
	1. Sistem menampilkan laman detail ajuan SKEM tanpa ada nama portofolio yang bisa dipilih di dalamnnya.

4.3.24. Memvalidasi SKEM

Tabel 25: Spesifikasi Use Case memvalidasi SKEM

Kode Use Case	UC024
Nama Use Case	Memvalidasi SKEM
Aktor	Validator SKEM
Deskripsi	Proses bagi validator SKEM untuk memvalidasi SKEM yang telah diajukan oleh mahasiswa
Kondisi Awal	Validator SKEM telah masuk ke dalam sistem dan membuka menu Validasi SKEM
Kondisi Akhir	Status ajuan SKEM menjadi ‘revisi’ atau ‘disetujui’
Alur Normal	
Aktor	Sistem
1. Validator SKEM memilih ajuan SKEM yang akan divalidasi	2. Sistem menampilkan detail ajuan SKEM yang dipilih

3. Validator SKEM mengisi form validasi dan menekan tombol simpan	
	4. Sistem menampilkan konfirmasi untuk memevalidasi SKEM
5. Validator SKEM menekan tombol ‘simpan’ E.2 Validator SKEM menekan tombol batal	
	6. Sistem merubah status ajuan SKEM menjadi ‘disetujui’ / ‘revisi’
Alur Eksepsi	
E.1. Validator SKEM menekan tombol batal	
Aktor	Sistem
	1. Sistem mengembalikan pengguna ke laman detail ajuan SKEM

4.3.25. Melihat perkembangan SKEM anak wali

Tabel 26: Spesifikasi Use Case melihat perkembangan SKEM anak wali

Kode Use Case	UC025
Nama Use Case	Melihat perkembangan SKEM anak wali
Aktor	Validator SKEM
Deskripsi	Proses bagi validator SKEM untuk melihat perkembangan SKEM anak Wali
Kondisi Awal	Validator SKEM telah masuk ke dalam sistem dan membuka menu Laporan SKEM
Kondisi Akhir	Laman berisi nilai-nilai SKEM anak wali yang dipilih setiap semesternya akan ditampilkan
Alur Normal	
Aktor	Sistem
1. Validator SKEM memilih anak wali yang ingin dilihat perkembangan SKEMnya	2. Sistem menampilkan detail ajuan SKEM anak Wali setiap semesternya beserta jumlah portofolio yang masih bisa diajukan pada semester ini

[Halaman ini sengaja dikosongkan]

BAB V
IMPLEMENTASI SISTEM

Bab ini membahas tentang implementasi dari sistem yang kami buat. Implementasi ini akan dibagi ke dalam beberapa bagian, yaitu bagian implementasi lapisan kontrol dan implementasi antarmuka pengguna.

5.1. Modul Portofolio

Implementasi lapisan kontrol ini berisi logika yang digunakan aplikasi, pada kasus kali untuk portofolio akan menggunakan kompetisi.

5.1.1. Kompetisi Controller

Lapisan ini bertugas untuk menghubungkan operasi CRUD (create, read, update, delete), mengajukan dan membatalkan ajuan portofolio dengan form yang ada pada view.

```
<?php
namespace App\Modules\Portofolio\Presentation\Controllers\K
ompetisi;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Modules\Portofolio\Presentation\Helpers\KompetisiHe
lper;
use App\Modules\Portofolio\Presentation\Requests\KompetisiR
equest;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Kom
petisi;
```

```
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Ska
laKompetisi;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Ang
gotaKompetisi;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Cap
aianKompetisi;
use App\Modules\Portofolio\Presentation\Controllers\Portofo
lioController;

class KompetisiController extends PortofolioController
{
    public function index(Request $request)
    {
        $this->authorize('viewAny', Kompetisi::class);

        $kompetisi = Kompetisi::whereHas('anggota', functio
n($query) {
            $query->where('id_mhs', auth()->user()-
>id_mhs);
        })->get();

        return view('Portofolio::kompetisi.index', compact(
'kompetisi'));
    }

    public function create()
    {
        $this->authorize('create', Kompetisi::class);

        $skala = SkalaKompetisi::active()->get();
        $capaian = CapaianKompetisi::active()->get();
```

```
        $luaran = Kompetisi::allJenisLuaran();

        return view('Portofolio::kompetisi.create', compact
('skala', 'capaian', 'luaran'));
    }

    public function store(KompetisiRequest $request)
    {
        $this->authorize('store', Kompetisi::class);

        try {
            $id_kompetisi = null;

            DB::transaction(function () use ($request, &$id
_kompetisi) {
                if ($request-
>skala_kompetisi == Kompetisi::SKALA_LOKAL) {
                    $request-
>jml_prov_negara_ikutserta = 1;
                }

                $kompetisi = Kompetisi::create([
                    'id_mhs'                        => auth
()->user()->id_mhs,
                    'nama'                          => $req
uest->nama,
                    'penyelenggara'                 => $req
uest->penyelenggara,
```

```
                    'url_kompetisi'                 => $req
uest->url_kompetisi,
                    'tgl_mulai'                     => $req
uest->tgl_mulai,
                    'tgl_selesai'                   => $req
uest->tgl_selesai,
                    'id_jenis_kompetisi'            => $req
uest->jenis_kompetisi,
                    'id_skala_kompetisi'            => $req
uest->skala_kompetisi,
                    'id_capaian_kompetisi'          => $req
uest->capaian_kompetisi,
                    'jenis_luaran'                  => $req
uest->jenis_luaran,
                    'luaran'                        => $req
uest->luaran,
                    'is_regu'                       => Komp
etisi::INITIAL_IS_REGU,
                    'is_bidang_ilmu_berhubungan'    => $req
uest->is_bidang_ilmu_berhubungan,
                    'jml_anggota'                   => Komp
etisi::INITIAL_JUMLAH_ANGGOTA,
                    'jml_peserta'                   => $req
uest->jml_peserta,
                    'jml_prov_negara_ikutserta'     => $req
uest->jml_prov_negara_ikutserta,
                    'updater'                       => auth
()->user()->id_user,
                ]);

                AnggotaKompetisi::create([
```

```
                    'id_mhs'        => auth()->user()-
>id_mhs,
                    'id_kompetisi'  => $kompetisi-
>id_kompetisi,
                    'peran'         => Kompetisi::PERAN_KET
UA,
                    'urutan'        => Kompetisi::INITIAL_U
RUTAN,
                    'updater'       => auth()->user()-
>id_user,
                ]);

                $id_kompetisi = $kompetisi->id_kompetisi;
            });

            return redirect()-
>route('Portofolio::kompetisi.show', ['id_kompetisi' => $id
_kompetisi])
                -
>with('success', __('Portofolio::general.success_simpan'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }

    public function edit(Request $request)
    {
```

```
        $kompetisi = Kompetisi::with([
            'capaian',
            'jenis',
            'skala'
        ])->findOrFail($request->id_kompetisi);

        $this->authorize('edit', $kompetisi);

        $skala = SkalaKompetisi::active()->get();
        $capaian = CapaianKompetisi::active()->get();
        $luaran = Kompetisi::allJenisLuaran();

        return view('Portofolio::kompetisi.edit', compact('
skala', 'capaian', 'luaran', 'kompetisi'));
    }

    public function update(KompetisiRequest $request)
    {
        try {
            $kompetisi = Kompetisi::findOrFail($request-
>id_kompetisi);

            $this->authorize('update', $kompetisi);

            if ($request-
>skala_kompetisi == Kompetisi::SKALA_LOKAL) {
                $request->jml_prov_negara_ikutserta = 1;
            }

            $kompetisi->update([
```

```
                'nama'                          => $request
->nama,
                'penyelenggara'                 => $request
->penyelenggara,
                'url_kompetisi'                 => $request
->url_kompetisi,
                'tgl_mulai'                     => $request
->tgl_mulai,
                'tgl_selesai'                   => $request
->tgl_selesai,
                'id_jenis_kompetisi'            => $request
->jenis_kompetisi,
                'id_skala_kompetisi'            => $request
->skala_kompetisi,
                'id_capaian_kompetisi'          => $request
->capaian_kompetisi,
                'jenis_luaran'                  => $request
->jenis_luaran,
                'luaran'                        => $request
->luaran,
                'is_bidang_ilmu_berhubungan'    => $request
->is_bidang_ilmu_berhubungan,
                'jml_peserta'                   => $request
->jml_peserta,
                'jml_prov_negara_ikutserta'     => $request
->jml_prov_negara_ikutserta,
                'updater'                       => auth()-
>user()->id_user,
            ]);
```

```
            return redirect()-
>route('Portofolio::kompetisi.show', ['id_kompetisi' => $re
quest->id_kompetisi])
                -
>with('success', __('Portofolio::general.success_perbarui')
);
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }

    public function show(Request $request)
    {
        $kompetisi = Kompetisi::with([
            'capaian',
            'jenis',
            'skala',
            'pembimbing',
            'anggota' => function ($query) {
                $query->orderBy('urutan');
            },
        ])->findOrFail($request->id_kompetisi);

        $this->authorize('view', $kompetisi);

        $mappedDokumen = KompetisiHelper::berkasMapper($req
uest->id_kompetisi);
```

```
        return view('Portofolio::kompetisi.show', compact('
kompetisi', 'mappedDokumen'));
    }

    public function delete(Request $request)
    {
        try {
            $kompetisi = Kompetisi::with([
                'anggota',
                'pembimbing'
            ])->findOrFail($request->id_kompetisi);

            $this->authorize('delete', $kompetisi);

            DB::transaction(function () use ($kompetisi) {
                foreach ($kompetisi->anggota as $anggota) {
                    $anggota->delete();
                }
                foreach ($kompetisi-
>pembimbing as $pembimbing) {
                    $pembimbing->delete();
                }
                foreach ($kompetisi->dokumen as $dokumen) {
                    $dokumen->delete();
                }
                $kompetisi->delete();
            });
```

```
            return redirect()-
>route('Portofolio::kompetisi')
                -
>with('success', __('Portofolio::general.success_hapus'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }

    public function submit(Request $request)
    {
        $kompetisi = Kompetisi::findOrFail($request-
>id_kompetisi);

        $this->authorize('submit', [$kompetisi, $request-
>is_dokumen_komplet]);

        $kompetisi->update([
            'status_validasi' => Kompetisi::STATUS_VALIDASI
_MENUNGGU
        ]);

        return redirect()-
>route('Portofolio::kompetisi.show', ['id_kompetisi' => $re
quest->id_kompetisi])
            -
>with('success', __('Portofolio::general.success_ajukan'));
    }
```

```
    public function cancel(Request $request)
    {
        $kompetisi = Kompetisi::findOrFail($request-
>id_kompetisi);

        $this->authorize('cancel', $kompetisi);

        $kompetisi->update([
            'status_validasi' => Kompetisi::STATUS_VALIDASI
_REVISI
        ]);

        return redirect()-
>route('Portofolio::kompetisi.show', ['id_kompetisi' => $re
quest->id_kompetisi])
            -
>with('success', __('Portofolio::general.success_batalkan')
);
    }
}
```

5.1.2. Berkas Kompetisi Controller

Lapisan ini bertugas untuk menghubungkan fitur manajemen berkas portofolio dengan database.

```
<?php

namespace App\Modules\Portofolio\Presentation\Controllers\K
ompetisi;
```

```
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Modules\Portofolio\Presentation\Helpers\KompetisiHe
lper;
use App\Modules\Portofolio\Presentation\Requests\UploadFile
Request;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Kom
petisi;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Dok
umenKompetisi;
use App\Modules\Portofolio\Presentation\Controllers\BaseBer
kasController;

class BerkasKompetisiController extends BaseBerkasControlle
r
{
    public function edit(Request $request)
    {
        $kompetisi = Kompetisi::findOrFail($request-
>id_kompetisi);

        $this->authorize('edit', $kompetisi);

        $mappedDokumen = KompetisiHelper::berkasMapper($req
uest->id_kompetisi);

        return view('Portofolio::kompetisi.berkas.edit', co
mpact('mappedDokumen', 'kompetisi'));
    }
```

```
    public function store(UploadFileRequest $request)
    {
        $id_dokumen = null;
        $upload = $this->upload($request, $id_dokumen);

        if ($upload == self::STATUS_SUCCESS) {

            $data = [
                'id_dokumen'                => $id_dokumen,
                'id_jenis_dok_portofolio'   => $request-
>id_jenis_dok_portofolio,
                'id_kompetisi'              => $request-
>id_kompetisi,
                'updater'                   => auth()-
>user()->id_user,
            ];

            DokumenKompetisi::create($data);

            $request->session()-
>flash('success', __('Portofolio::berkas.success_simpan'));

            return response()-
>json(['success' => self::STATUS_SUCCESS]);
        } else {
            return response()-
>json(['errors' => ['general' => [__('Portofolio::general.e
rror')]]], 500);
        }
```

```
    }

    public function delete(Request $request)
    {
        try {
            DokumenKompetisi::where([
                'id_dokumen'                => $request-
>id_dokumen,
                'id_jenis_dok_portofolio'   => $request-
>id_jenis_dok_portofolio,
                'id_kompetisi'              => $request-
>id_kompetisi
            ])->delete();

            return redirect()-
>route('Portofolio::kompetisi.berkas.edit', ['id_kompetisi'
 => $request->id_kompetisi])
                -
>with('success', __('Portofolio::berkas.success_hapus'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }

    public function select(Request $request)
    {
        try {
            DB::transaction(function () use ($request) {
```

```
                foreach ($request-
>id_dokumen as $id_dokumen) {
                    DokumenKompetisi::create([
                        'id_dokumen'                => $id_
dokumen,
                        'id_jenis_dok_portofolio'   => $req
uest->id_jenis_dok_portofolio,
                        'id_kompetisi'              => $req
uest->id_kompetisi,
                        'updater'                   => auth
()->user()->id_user,
                    ]);
                }
            });

            return redirect()-
>route('Portofolio::kompetisi.berkas.edit', ['id_kompetisi'
 => $request->id_kompetisi])
                -
>with('success', __('Portofolio::berkas.success_pilih'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }
}
```

5.1.3. Anggota Kompetisi Controller

Lapisan ini bertugas untuk menghubungkan operasi yang berhubungan dengan anggota pada portofolio dan form yang yang ada di view.

```
<?php

namespace App\Modules\Portofolio\Presentation\Controllers\K
ompetisi;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Kom
petisi;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Ang
gotaKompetisi;
use App\Modules\Portofolio\Presentation\Controllers\Portofo
lioController;

class AnggotaKompetisiController extends PortofolioControll
er
{
    public function edit(Request $request)
    {
        $kompetisi = Kompetisi::find($request-
>id_kompetisi);

        $this->authorize('edit', $kompetisi);

        $anggota = AnggotaKompetisi::where('id_kompetisi',
$request->id_kompetisi)
            ->orderBy('urutan')->get();
```

```
        $id_kompetisi = $request->id_kompetisi;

        return view('Portofolio::kompetisi.anggota.edit', c
ompact('anggota', 'id_kompetisi'));
    }

    public function store(Request $request)
    {
        try {
            $kompetisi = Kompetisi::find($request-
>id_kompetisi);

            $this->authorize('edit', $kompetisi);

            DB::transaction(function () use ($request, $kom
petisi) {
                foreach ($request->id_anggota as $id_mhs) {
                    $kompetisi->jml_anggota++;
                    AnggotaKompetisi::create([
                        'id_mhs'        => $id_mhs,
                        'id_kompetisi'  => $request-
>id_kompetisi,
                        'peran'         => Kompetisi::PERAN
_ANGGOTA,
                        'urutan'        => $kompetisi-
>jml_anggota,
                        'updater'       => auth()->user()-
>id_user,
```

```
                    ]);
                }
                $kompetisi->is_regu = 1;
                $kompetisi->update();
            });

            return redirect()-
>route('Portofolio::kompetisi.anggota.edit', ['id_kompetisi
' => $request->id_kompetisi])
                -
>with('success', __('Portofolio::kompetisi.sukses_tambah_an
ggota'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }

    public function delete(Request $request)
    {
        try {
            $kompetisi = Kompetisi::with([
                'anggota' => function ($q) {
                    $q-
>where('peran', Kompetisi::PERAN_ANGGOTA)-
>orderBy('urutan', 'desc');
                }])->findOrFail($request->id_kompetisi);

            $this->authorize('edit', $kompetisi);
```

```
            DB::transaction(function () use ($request, $kom
petisi) {
                $kompetisi->jml_anggota--;
                if($kompetisi->jml_anggota == 1) {
                    $kompetisi->is_regu = 0;
                }
                $kompetisi->update();
                foreach ($kompetisi->anggota as $anggota) {
                    if(
                        $anggota-
>id_mhs        == $request->id_mhs &&
                        $anggota-
>id_kompetisi  == $request->id_kompetisi
                    ) {
                        $anggota->delete();
                    } else {
                        $anggota->urutan = $kompetisi-
>jml_anggota--;
                        $anggota->update();
                    }
                }
            });

            return redirect()-
>route('Portofolio::kompetisi.anggota.edit', ['id_kompetisi
' => $request->id_kompetisi])
                -
>with('success', __('Portofolio::kompetisi.sukses_hapus_ang
gota'));
```

```
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
    }
}
```

5.1.4. Pembimbing Kompetisi Controller

Lapisan ini bertugas untuk menghubungkan operasi yang berhubungan dengan pembimbing pada portofolio dan form yang yang ada di view.

```
<?php

namespace App\Modules\Portofolio\Presentation\Controllers\K
ompetisi;

use App\Common\Model\Sdm;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Kom
petisi;
use App\Modules\Portofolio\Presentation\Controllers\Portofo
lioController;
use App\Modules\Portofolio\Presentation\Model\Kompetisi\Pem
bimbingKompetisi;
use App\Modules\Portofolio\Presentation\Requests\Pembimbing
KompetisiRequest;
```

```
class PembimbingKompetisiController extends PortofolioContr
oller
{
    public function edit(Request $request)
    {
        $kompetisi = Kompetisi::find($request-
>id_kompetisi);

        $this->authorize('edit', $kompetisi);

        $pembimbing = PembimbingKompetisi::with('sdm')-
>where('id_kompetisi', $request->id_kompetisi)->get();

        $id_kompetisi = $request->id_kompetisi;

        return view('Portofolio::kompetisi.pembimbing.edit'
, compact('pembimbing', 'id_kompetisi'));
    }

    public function store(PembimbingKompetisiRequest $reque
st)
    {
        try {
            $kompetisi = Kompetisi::find($request-
>id_kompetisi);

            $this->authorize('edit', $kompetisi);

            DB::transaction(function () use ($request) {
```

```
                foreach ($request-
>id_pembimbing as $id_orang_sdm) {
                    $pembimbing = Sdm::findOrFail($id_orang
_sdm);
                    PembimbingKompetisi::create([
                        'id_orang_sdm'  => $id_orang_sdm,
                        'id_kompetisi'  => $request-
>id_kompetisi,
                        'nip'           => $pembimbing-
>nip,
                        'nidn'          => $pembimbing-
>nidn,
                        'nama'          => $pembimbing-
>nama,
                        'is_eksternal'  => 0,
                        'updater'       => auth()->user()-
>id_user,
                    ]);
                }
            });

            return redirect()-
>route('Portofolio::kompetisi.pembimbing.edit', ['id_kompet
isi' => $request->id_kompetisi])
                -
>with('success', __('Portofolio::kompetisi.sukses_tambah_pe
mbimbing'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
```

```
        }
    }

    public function delete(Request $request)
    {
        try {
            $kompetisi = Kompetisi::find($request-
>id_kompetisi);

            $this->authorize('edit', $kompetisi);

            PembimbingKompetisi::where([
                'id_pembimbing_kompetisi'    => $request-
>id_pembimbing_kompetisi,
                'id_kompetisi'               => $request-
>id_kompetisi
            ])->delete();

            return redirect()-
>route('Portofolio::kompetisi.pembimbing.edit', ['id_kompet
isi' => $request->id_kompetisi])
                -
>with('success', __('Portofolio::kompetisi.sukses_hapus_pem
bimbing'));
        } catch (\Exception $e) {
            return redirect()->back()
                -
>with('error', __('Portofolio::general.error'));
        }
```

```
    }
}
```

5.1.5. Berkas Controller

Lapisan ini bertugas untuk menghubungkan fitur manajemen berkas secara umum dengan database.

```
<?php

namespace App\Modules\Portofolio\Presentation\Controllers;

use Illuminate\Http\Request;
use App\Modules\Portofolio\Presentation\Helpers\FileStorage
;
use App\Modules\Portofolio\Presentation\Model\Dokumen\Dokum
en;
use App\Modules\Portofolio\Presentation\Model\Dokumen\Jenis
Dokumen;
use App\Modules\Portofolio\Presentation\Requests\UploadFile
Request;

class BerkasController extends BaseBerkasController
{
    public const STATUS_USED   = '23000';
    public function index()
    {
        $jenisDokumen = JenisDokumen::active()->get();

        return view('Portofolio::berkas.index', compact('je
nisDokumen'));
```

```
    }

    public function store(UploadFileRequest $request)
    {
        $id_dokumen = null;
        $upload = $this->upload($request, $id_dokumen);

        if ($upload == self::STATUS_SUCCESS) {

            return response()-
>json(['success' => 'Berkas berhasil disimpan!']);
        } else {

            return response()-
>json(['error' => __('Portofolio::general.error')]);
        }
    }

    public function update(UploadFileRequest $request)
    {
        try {
            $dokumen = Dokumen::find($request->id_dokumen);
            $data = [
                "nama_file"     => $request->nama_file,
                "keterangan"    => $request->keterangan,
                "updater"       => auth()->user()->id_user,
            ];

            $dokumen->update($data);
```

```
            return response()-
>json(['success' => 'Berkas berhasil diperbarui!']);

        } catch (\Exception $e) {

            return response()-
>json(['error' => __('Portofolio::general.error')]);
        }
    }

    public function delete(Request $request)
    {
        try {
            $dokumen = Dokumen::findOrFail($request-
>id_dokumen);

            // check document exist and document uploader
            if($dokumen && ($dokumen->id_mhs == auth()-
>user()->id_mhs)) {

                $dokumen->delete();

                FileStorage::delete($dokumen->file_id);

                return response()-
>json(['success' => 'Berkas berhasil dihapus!']);
            } else {

                return response()-
>json(['error' => 'Berkas tidak ditemukan!']);
```

```
            }

        } catch (\Exception $e) {

            if($e->getCode() == self::STATUS_USED) {

                return response()-
>json(['error' => 'Gagal menghapus karena berkas sedang dig
unakan di portofolio.']);
            }

            return response()-
>json(['error' => __('Portofolio::general.error')]);
        }
    }

    public function stream(Request $request)
    {
        try {
            $dokumen = Dokumen::find($request->id_dokumen);

            if($dokumen) {
                $b64 = file_get_contents(config('filestorag
e.base_uri') . $dokumen->public_link);

                file_put_contents($dokumen-
>nama_file, $b64);
                if($request->has('detail')) {
```

```
                    return response()->file($dokumen-
>nama_file)->deleteFileAfterSend(true);
                }
                if ($request->has('unduh')) {
                    return response()->download($dokumen-
>nama_file, $dokumen->nama_file .  '.' . $dokumen-
>ekstensi)->deleteFileAfterSend(true);
                }

            }

            $request->session()-
>flash('error', 'Berkas tidak ditemukan!');

            return redirect()->route('Portofolio::berkas');
        } catch (\Exception $e) {

            $request->session()-
>flash('error', __('Portofolio::general.error'));

            return redirect()->route('Portofolio::berkas');
        }
    }
}
```

5.2 Modul SKEM

5.2.1 Controller

5.2.1.1 Mahasiswa Controller

```
<?php

namespace
App\Modules\Skem\Presentation\Controllers;

use App\Http\Controllers\Controller;

use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlAbdimasMagangRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlAbdimasWirausahaRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlKegiatanPelatihanRepository;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Session;

use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlMahasiswaRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlSemesterRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlSkemRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlKompetisiRepository;
```

```
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlWirausahaRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlOrmawaRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlKegiatanRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlLkmmPemanduRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlPelatihanRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlAbdimasRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlMagangRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlInternasionalisasiRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlPertukaranMhsRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlInternasionalisasiPelatihanRepository;
use
App\Modules\Skem\Infrastructure\Persistence\Ms
sqlKaryaIlmiahRepository;

use
App\Modules\Skem\Application\Mahasiswa\DeleteS
kem\DeleteSkemService;
```

```
use
App\Modules\Skem\Application\Mahasiswa\StoreSk
em\StoreSkemService;
use
App\Modules\Skem\Application\Mahasiswa\StoreSk
em\StoreSkemRequest;

use
App\Modules\Skem\Application\Mahasiswa\UpdateS
kem\UpdateSkemService;
use
App\Modules\Skem\Application\Mahasiswa\UpdateS
kem\UpdateSkemRequest;

use
App\Modules\Skem\Application\Mahasiswa\ShowSke
m\ShowSkemService;

use
App\Modules\Skem\Application\Mahasiswa\ShowKeg
iatanSkem\ShowKegiatanSkemService;


use
App\Modules\Skem\Application\Mahasiswa\CreateS
kem\CreateSkemService;
use
App\Modules\Skem\Application\Mahasiswa\CreateS
kem\CreateSkemRequest;

use
App\Modules\Skem\Application\Mahasiswa\EditSke
m\EditSkemService;
use
App\Modules\Skem\Application\Mahasiswa\EditSke
m\EditSkemRequest;
```

```
use
App\Modules\Skem\Application\Mahasiswa\IndexSk
em\IndexSkemService;
use
App\Modules\Skem\Application\Mahasiswa\IndexSk
em\IndexSkemResponse;

use
App\Modules\Skem\Application\Mahasiswa\UpdateS
tatusSkem\UpdateStatusSkemService;
use
App\Modules\Skem\Application\Mahasiswa\UpdateS
tatusSkem\UpdateStatusSkemRequest;

class MahasiswaController extends Controller
{
    protected $createSkemService;
    protected $indexSkemService;
    protected $editSkemService;
    protected $updateSkemService;
    protected $updateStatusSkemService;
    protected $storeSkemService;
    protected $showSkemService;

    public function __construct()
    {
        $this->indexSkemService = new
IndexSkemService(
            new MssqlMahasiswaRepository(),
            new MssqlSkemRepository()
        );

        $this->storeSkemService = new
StoreSkemService(
            new MssqlMahasiswaRepository(),
            new MssqlSemesterRepository(),
            new MssqlSkemRepository(),
            new MssqlKompetisiRepository(),
            new MssqlWirausahaRepository(),
```

```
            new MssqlMagangRepository(),
            new MssqlOrmawaRepository(),
            new MssqlKegiatanRepository(),
            new
MssqlKegiatanPelatihanRepository(),
            new MssqlLkmmPemanduRepository(),
            new MssqlAbdimasRepository(),
            new
MssqlAbdimasWirausahaRepository(),
            new
MssqlAbdimasMagangRepository(),
            new MssqlPelatihanRepository(),
            new
MssqlInternasionalisasiRepository(),
            new
MssqlPertukaranMhsRepository(),

            new
MssqlInternasionalisasiPelatihanRepository()
        );

        $this->updateSkemService = new
UpdateSkemService(
            new MssqlMahasiswaRepository(),
            new MssqlSemesterRepository(),
            new MssqlSkemRepository(),
            new MssqlKompetisiRepository(),
            new MssqlWirausahaRepository(),
            new MssqlMagangRepository(),
            new MssqlOrmawaRepository(),
            new MssqlKegiatanRepository(),
            new
MssqlKegiatanPelatihanRepository(),
            new MssqlLkmmPemanduRepository(),
            new MssqlAbdimasRepository(),
            new
MssqlAbdimasWirausahaRepository(),
```

```
            new
MssqlAbdimasMagangRepository(),
            new MssqlPelatihanRepository(),
            new
MssqlInternasionalisasiRepository(),
            new
MssqlPertukaranMhsRepository(),

            new
MssqlInternasionalisasiPelatihanRepository()
        );

        $this->showSkemService = new
ShowSkemService(
            new MssqlSkemRepository()
        );

        $this->showKegiatanSkemService = new
ShowKegiatanSkemService(
            new MssqlSkemRepository()
        );

        $this->deleteSkemService = new
DeleteSkemService(
            new MssqlSkemRepository(),
            new MssqlSemesterRepository(),
            new MssqlKompetisiRepository(),
            new MssqlWirausahaRepository(),
            new MssqlMagangRepository(),
            new MssqlOrmawaRepository(),
            new MssqlKegiatanRepository(),
            new MssqlLkmmPemanduRepository(),
            new MssqlPelatihanRepository(),
            new MssqlAbdimasRepository(),
            new
MssqlInternasionalisasiRepository(),
            new MssqlPertukaranMhsRepository()
        );
```

```
            $this->createSkemService = new
    CreateSkemService(
                new MssqlMahasiswaRepository(),
                new MssqlKompetisiRepository(),
                new MssqlWirausahaRepository(),
                new MssqlMagangRepository(),
                new MssqlOrmawaRepository(),
                new MssqlKegiatanRepository(),
                new
    MssqlKegiatanPelatihanRepository(),
                new MssqlLkmmPemanduRepository(),
                new MssqlAbdimasRepository(),
                new
    MssqlAbdimasWirausahaRepository(),
                new
    MssqlAbdimasMagangRepository(),
                new MssqlPelatihanRepository(),
                new
    MssqlInternasionalisasiRepository(),
                new
    MssqlPertukaranMhsRepository(),

                new
    MssqlInternasionalisasiPelatihanRepository(),
                new MssqlKaryaIlmiahRepository()
            );

            $this->editSkemService = new
    EditSkemService(
                new MssqlMahasiswaRepository(),
                new MssqlSemesterRepository(),
                new MssqlSkemRepository(),
                new MssqlKompetisiRepository(),
                new MssqlWirausahaRepository(),
                new MssqlMagangRepository(),
                new MssqlOrmawaRepository(),
                new MssqlKegiatanRepository(),
```

```
            new
MssqlKegiatanPelatihanRepository(),
            new MssqlLkmmPemanduRepository(),
            new MssqlAbdimasRepository(),
            new
MssqlAbdimasWirausahaRepository(),
            new
MssqlAbdimasMagangRepository(),
            new MssqlPelatihanRepository(),
            new
MssqlInternasionalisasiRepository(),
            new
MssqlPertukaranMhsRepository(),

            new
MssqlInternasionalisasiPelatihanRepository()
        );

        $this->updateStatusSkemService = new
UpdateStatusSkemService(
            new MssqlSkemRepository()
        );
    }

    public function index()
    {
        $idMhs = Auth::user()->id_mhs;
        $response = $this->indexSkemService-
>handle($idMhs);
//        dd($response);
        return view('Skem::mahasiswa.index',
compact('response'));
    }

    public function show(Request $request)
    {
//        dd($request);
        $idSkem = $request->idSkem;
```

```
        $response = $this->showSkemService-
>handle($idSkem);
//        dd($response);
        return view('Skem::mahasiswa.show',
compact('response'));
    }

    public function create(Request $request)
    {
//        dd($request);
        $createSkemRequest = new
CreateSkemRequest(Auth::user()->id_mhs,
$request->idSmt);
        $response = $this->createSkemService-
>handle($createSkemRequest);
        return view('Skem::mahasiswa.create',
compact('response'));
    }

    public function store(Request $request)
    {
        $storeSkemRequest = new
StoreSkemRequest(
            $request->idSmt,
            $request->user()->id_mhs,
            $request->kompetisi,
            $request->wirausaha,
            $request->magang,
            $request->ormawa,
            $request->kegiatan,
            $request->kegiatanPelatihan,
            $request->lkmmPemandu,
            $request->pelatihan,
            $request->abdimas,
            $request->abdimasWirausaha,
            $request->abdimasMagang,
            $request->internasionalisasi,
```

```
                $request->pertukaranMhs,

                $request-
>internasionalisasiPelatihan
            ) ;
//            dd($storeSkemRequest);
            $response = $this->storeSkemService-
>handle($storeSkemRequest);
            return redirect()-
>route('Skem::mahasiswa.show', ['idSkem' =>
$response]);
        }

        public function edit(Request $request)
        {
            $editSkemRequest = new
EditSkemRequest(Auth::user()->id_mhs,
$request->idSmt, $request->idSkem);
//            dd($request);
            $response = $this->editSkemService-
>handle($editSkemRequest);
            return view('Skem::mahasiswa.edit',
compact('response'));
        }

        public function update(Request $request)
        {
            $idMhs = Auth::user()->id_mhs;
            $request = new UpdateSkemRequest(
                $idMhs,
                $request->idSkem,
                $request->kompetisi,
                $request->wirausaha,
                $request->magang,
                $request->ormawa,
                $request->kegiatan,
                $request->kegiatanPelatihan,
                $request->lkmmPemandu,
                $request->pelatihan,
```

```
                $request->abdimas,
                $request->abdimasWirausaha,
                $request->abdimasMagang,
                $request->internasionalisasi,
                $request->pertukaranMhs,

                $request-
>internasionalisasiPelatihan
            ) ;
//            dd($request);
            $response = $this->updateSkemService-
>handle($request);
            return redirect()-
>route('Skem::mahasiswa.show',['idSkem' =>
$response]);

        }

        public function delete()
        {
            $requests = [
                'FE4E773A-DE68-4C59-BC4A-
A110927CA8C4',
                '5C6A91F4-7447-4611-A466-
84D05FF7D0D4',
                '4E81DB1C-1949-4030-B5C9-
71E7B55F08C9',
                'EE24D988-BD20-4FF5-AA43-
24B0B9F65312'
                ];

            foreach ($requests as $request){
                $response = $this-
>deleteSkemService->handle($request);
            }

        }
```

```
    public function kegiatan(Request $request)
    {
        $idKegiatanSkem = $request->idKegiatanSkem;
//        $idMhs = Auth::user()->id_mhs;
        $response = $this->showKegiatanSkemService->handle($idKegiatanSkem);
        return view('Skem::mahasiswa.kegiatan', compact('response'));

    }

    public function status(Request $request)
    {
        return redirect()->route('Skem::mahasiswa.show', ['idSkem'=>$request->idSkem])->with('error', 'Saat ini belum memasuki rentang waktu untuk mengajukan SKEM');

        $request = new UpdateStatusSkemRequest($request->idSkem, 1, Auth::user()->id_mhs);
        $response = $this->updateStatusSkemService->handle($request);
        return redirect()->route('Skem::mahasiswa.show', ['idSkem' => $response]);
    }

}
```

5.2.1.2 Validator Controller

```
<?php

namespace App\Modules\Skem\Presentation\Controllers;
```

```
use App\Http\Controllers\Controller;


use
App\Modules\Skem\Application\Validator\IndexPerkembanganSkem\Ind
exPerkembanganSkemService;
use
App\Modules\Skem\Application\Validator\DetailPerkembanganSkem\De
tailPerkembanganSkemService;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlAbdimasMagangRe
pository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlAbdimasWirausaha
Repository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlKegiatanPelatihanR
epository;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Session;

use
App\Modules\Skem\Infrastructure\Persistence\MssqlMahasiswaReposito
ry;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlSemesterRepository
;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlSkemRepository;

use
App\Modules\Skem\Infrastructure\Persistence\MssqlKompetisiRepositor
y;
```

```
use
App\Modules\Skem\Infrastructure\Persistence\MssqlWirausahaRepositor
y;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlOrmawaRepository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlKegiatanRepository
;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlLkmmPemanduRep
ository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlPelatihanRepository
;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlAbdimasRepository
;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlMagangRepository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlInternasionalisasiRe
pository;
use
App\Modules\Skem\Infrastructure\Persistence\MssqlPertukaranMhsRepo
sitory;



use
App\Modules\Skem\Application\Validator\ShowSkem\ShowSkemServic
e;
use
App\Modules\Skem\Application\Validator\ShowKegiatanSkem\ShowKe
giatanSkemService;
use
App\Modules\Skem\Application\Validator\IndexSkem\IndexSkemServic
e;
```

```
use
App\Modules\Skem\Application\Validator\IndexSkem\IndexSkemRespo
nse;

use
App\Modules\Skem\Application\Validator\UpdateStatusSkem\UpdateSta
tusSkemService;
use
App\Modules\Skem\Application\Validator\UpdateStatusSkem\UpdateSta
tusSkemRequest;

class ValidatorController extends Controller
{
   protected $indexPerkembanganSkemService;
   protected $detailPerkembanganSkemService;

   protected $indexSkemService;
   protected $updateStatusSkemService;
   protected $showKegiatanService;
   protected $showSkemService;
   protected $showKegiatanSkemService;

   public function __construct()
   {
      $this->indexPerkembanganSkemService = new
IndexPerkembanganSkemService(
         new MssqlMahasiswaRepository()
      );

      $this->detailPerkembanganSkemService = new
DetailPerkembanganSkemService(
         new MssqlMahasiswaRepository(),
         new MssqlSemesterRepository(),
         new MssqlSkemRepository(),
         new MssqlKompetisiRepository(),
         new MssqlWirausahaRepository(),
         new MssqlMagangRepository(),
```

```
        new MssqlOrmawaRepository(),
        new MssqlKegiatanRepository(),
        new MssqlKegiatanPelatihanRepository(),
        new MssqlLkmmPemanduRepository(),
        new MssqlAbdimasRepository(),
        new MssqlAbdimasWirausahaRepository(),
        new MssqlAbdimasMagangRepository(),
        new MssqlPelatihanRepository(),
        new MssqlInternasionalisasiRepository(),
        new MssqlPertukaranMhsRepository()
    );

    $this->indexSkemService = new IndexSkemService(
        new MssqlMahasiswaRepository(),
        new MssqlSemesterRepository(),
        new MssqlSkemRepository()
    );


    $this->showSkemService = new ShowSkemService(
        new MssqlMahasiswaRepository(),
        new MssqlSemesterRepository(),
        new MssqlSkemRepository(),
        new MssqlKompetisiRepository(),
        new MssqlWirausahaRepository(),
        new MssqlMagangRepository(),
        new MssqlOrmawaRepository(),
        new MssqlKegiatanRepository(),
        new MssqlKegiatanPelatihanRepository(),
        new MssqlLkmmPemanduRepository(),
        new MssqlAbdimasRepository(),
        new MssqlAbdimasWirausahaRepository(),
        new MssqlAbdimasMagangRepository(),
        new MssqlPelatihanRepository(),
        new MssqlInternasionalisasiRepository(),
        new MssqlPertukaranMhsRepository()
    );
```

```
        $this->showKegiatanSkemService = new
ShowKegiatanSkemService(
            new MssqlSemesterRepository(),
            new MssqlSkemRepository(),
            new MssqlKompetisiRepository(),
            new MssqlWirausahaRepository(),
            new MssqlMagangRepository(),
            new MssqlOrmawaRepository(),
            new MssqlKegiatanRepository(),
            new MssqlLkmmPemanduRepository(),
            new MssqlPelatihanRepository(),
            new MssqlAbdimasRepository(),
            new MssqlInternasionalisasiRepository(),
            new MssqlPertukaranMhsRepository()
        );

        $this->updateStatusSkemService = new UpdateStatusSkemService(
            new MssqlSkemRepository()
        );
    }

    public function indexPerkembangan(Request $request)
    {
        $idSdm = Auth::user()->id_sdm;
//        $request = $idSdm;
        $response = $this->indexPerkembanganSkemService-
>handle($idSdm);
        return view('Skem::validator.index-perkembangan',
compact('response'));
    }

    public function perkembangan(Request $request)
    {
        $idMhs = $request->idMhs;
//        dd($request);
        $response = $this->detailPerkembanganSkemService-
>handle($idMhs);
```

```
        return view('Skem::validator.perkembangan', compact('response'));
    }

    public function index(Request $request)
    {
        $idSdm = Auth::user()->id_sdm;
        $idSmt = $request->idSmt == null ? null : $request->idSmt;
//        $request = $idSdm;
        $response = $this->indexSkemService->handle($idSdm, $idSmt);
        return view('Skem::validator.index', compact('response'));
    }

    public function show(Request $request)
    {
        $requester = $request->requester;
        if ($requester == null) {
            $requester = 'perkembangan';
        }
        $idMhs = $request->idMhs;
        $idSmt = $request->idSmt;
        $idSkem = $request->idSkem;
        $response = $this->showSkemService->handle($idMhs, $idSmt,
$idSkem);
//        dd($response);
        return view('Skem::validator.show', compact('response', 'requester'));
    }

    public function kegiatan(Request $request)
    {
        $response = $this->showKegiatanSkemService->handle($request-
>idKegiatanSkem);
        return view('Skem::validator.kegiatan', compact('response'));

    }

    public function status(Request $request)
    {
        if ($request->statusAjuan == null) {
            $request->statusAjuan = 3;
```

```
        }
        $newRequest = new UpdateStatusSkemRequest($request->idSkem,
$request->statusAjuan
            , Auth::user()->id_sdm, $request->message);
//        dd($newRequest);
        $response = $this->updateStatusSkemService-
>handle($newRequest);
        return redirect()->route('Skem::validator.show', ['idSkem' =>
$request->idSkem, 'idMhs' => $request->idMhs, 'idSmt' => $request-
>idSmt]);
    }

}
```

5.2.2 Application

5.2.2.1 Mahasiswa CreateSkemRequest

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\CreateSkem;

class CreateSkemRequest
{
    public $idMhs;
    public $idSmt;

    public function __construct($idMhs, $idSmt)
    {
        $this->idMhs = $idMhs;
        $this->idSmt = $idSmt;
    }
}
```

5.2.2.2 Mahasiswa CreateSkemService

```
<?php
```

```
namespace App\Modules\Skem\Application\Mahasiswa\CreateSkem;

use
App\Modules\Skem\Application\Mahasiswa\CreateSkem\CreateSkemRe
quest;
use
App\Modules\Skem\Application\Mahasiswa\CreateSkem\CreateSkemRe
sponse;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use
App\Modules\Skem\Domain\Repositories\AbdimasMagangRepository;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;
use
App\Modules\Skem\Domain\Repositories\AbdimasWirausahaRepository
;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;
use
App\Modules\Skem\Domain\Repositories\KegiatanPelatihanRepository;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;
use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;
use App\Modules\Skem\Domain\Repositories\MagangRepository;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\OrmawaRepository;
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;
use App\Modules\Skem\Domain\Repositories\SemesterRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;

use
App\Modules\Skem\Domain\Repositories\InternasionalisasiPelatihanRep
ository;
use App\Modules\Skem\Domain\Repositories\KaryaIlmiahRepository;

class CreateSkemService
{
```

```
    private $mahasiswaRepository;
    private $semesterRepository;
    private $kompetisiRepository;
    private $wirausahaRepository;
    private $magangRepository;
    private $ormawaRepository;
    private $kegiatanRepository;
    private $kegiatanPelatihanRepository;
    private $lkmmPemanduRepository;
    private $abdimasRepository;
    private $abdimasWirausahaRepository;
    private $abdimasMagangRepository;
    private $pelatihanRepository;
    private $internasionalisasiRepository;
    private $pertukaranMhsRepository;

    private $internasionalisasiPelatihanRepository;
    private $karyaIlmiahRepository;

    public function __construct(
        MahasiswaRepository $mahasiswaRepository,
        KompetisiRepository $kompetisiRepository,
        WirausahaRepository $wirausahaRepository,
        MagangRepository $magangRepository,
        OrmawaRepository $ormawaRepository,
        KegiatanRepository $kegiatanRepository,
        KegiatanPelatihanRepository $kegiatanPelatihanRepository,
        LkmmPemanduRepository $lkmmPemanduRepository,
        AbdimasRepository $abdimasRepository,
        AbdimasWirausahaRepository $abdimasWirausahaRepository,
        AbdimasMagangRepository $abdimasMagangRepository,
        PelatihanRepository $pelatihanRepository,
        InternasionalisasiRepository $internasionalisasiRepository,
        PertukaranMhsRepository $pertukaranMhsRepository,

        InternasionalisasiPelatihanRepository
$internasionalisasiPelatihanRepository,
```

```
        KaryaIlmiahRepository $karyaIlmiahRepository
    )
    {
        $this->mahasiswaRepository = $mahasiswaRepository;
        $this->kompetisiRepository = $kompetisiRepository;
        $this->wirausahaRepository = $wirausahaRepository;
        $this->magangRepository = $magangRepository;
        $this->ormawaRepository = $ormawaRepository;
        $this->kegiatanRepository = $kegiatanRepository;
        $this->kegiatanPelatihanRepository                    =
$kegiatanPelatihanRepository;
        $this->lkmmPemanduRepository = $lkmmPemanduRepository;
        $this->abdimasRepository = $abdimasRepository;
        $this->abdimasWirausahaRepository                     =
$abdimasWirausahaRepository;
        $this->abdimasMagangRepository = $abdimasMagangRepository;
        $this->pelatihanRepository = $pelatihanRepository;
        $this->internasionalisasiRepository = $internasionalisasiRepository;
        $this->pertukaranMhsRepository = $pertukaranMhsRepository;

        $this->internasionalisasiPelatihanRepository          =
$internasionalisasiPelatihanRepository;
        $this->karyaIlmiahRepository = $karyaIlmiahRepository;
    }

    public function handle(CreateSkemRequest $createSkemRequest)
    {

        $idMhs = new MahasiswaId($createSkemRequest->idMhs);
        $idSmt = new SemesterId($createSkemRequest->idSmt);
        $mahasiswa = $this->mahasiswaRepository->get($idMhs);
        $semester = $mahasiswa->getSemester($idSmt);
        $periode = $mahasiswa->getThnAngkatan() == 2018 ? 21557 : 366;

        $kueriKumpulanKompetisi     =     $this->kompetisiRepository-
>requestList($idMhs, $periode);
        $kueriKumpulanWirausaha     =     $this->wirausahaRepository-
>requestList($idMhs, $periode);
```

```
        $kueriKumpulanMagang = $this->magangRepository->requestList($idMhs, $periode);

        $kueriKumpulanOrmawa = $this->ormawaRepository->requestList($idMhs, $periode);
        $kueriKumpulanKegiatan = $this->kegiatanRepository->requestList($idMhs, $periode);
        $kueriKumpulanKegiatanPelatihan = $this->kegiatanPelatihanRepository->requestList($idMhs, $periode);
        $kueriKumpulanLkmmPemandu = $this->lkmmPemanduRepository->requestList($idMhs, $periode);
        $kueriKumpulanPelatihan = $this->pelatihanRepository->requestList($idMhs, $periode);

        $kueriKumpulanAbdimas = $this->abdimasRepository->requestList($idMhs, $periode);
        $kueriKumpulanAbdimasWirausaha = $this->abdimasWirausahaRepository->requestList($idMhs, $periode);
        $kueriKumpulanAbdimasMagang = $this->abdimasMagangRepository->requestList($idMhs, $periode);

        $kueriKumpulanInternasionalisasi = $this->internasionalisasiRepository->requestList($idMhs, $periode);
        $kueriKumpulanPertukaranMhs = $this->pertukaranMhsRepository->requestList($idMhs, $periode);

        $kueriKumpulanInternasionalisasiPelatihan = $this->internasionalisasiPelatihanRepository->requestList($idMhs, $periode);
        $kueriKumpulanKaryaIlmiah = $this->karyaIlmiahRepository->requestList($idMhs, $periode);

        $response = new CreateSkemResponse(
            $kueriKumpulanKompetisi,
            $kueriKumpulanWirausaha,
            $kueriKumpulanMagang,
            $kueriKumpulanOrmawa,
            $kueriKumpulanKegiatan,
```

```
            $kueriKumpulanKegiatanPelatihan,
            $kueriKumpulanLkmmPemandu,
            $kueriKumpulanPelatihan,
            $kueriKumpulanAbdimas,
            $kueriKumpulanAbdimasWirausaha,
            $kueriKumpulanAbdimasMagang,
            $kueriKumpulanInternasionalisasi,
            $kueriKumpulanPertukaranMhs,

            $kueriKumpulanInternasionalisasiPelatihan,
            $kueriKumpulanKaryaIlmiah,
            $semester
        );

//      dd($response);
        return $response;
    }
}
```

5.2.2.3 Mahasiswa CreateSkemResponse

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\CreateSkem;

class CreateSkemResponse
{
    public $kumpulanKompetisi;
    public $kumpulanWirausaha;
    public $kumpulanMagang;
    public $kumpulanOrmawa;
    public $kumpulanKegiatan;
    public $kumpulanKegiatanPelatihan;
    public $kumpulanLkmmPemandu;
    public $kumpulanPelatihan;
    public $kumpulanAbdimas;
    public $kumpulanAbdimasWirausaha;
    public $kumpulanAbdimasMagang;
    public $kumpulanInternasionalisasi;
```

```
    public $kumpulanPertukaranMhs;

    public $kumpulanInternasionalisasiPelatihan;
    public $kumpulanKaryaIlmiah;

    public $idSmt;
    public $semesterKe;

    public function __construct(
        $kumpulanKompetisi,
        $kumpulanWirausaha,
        $kumpulanMagang,
        $kumpulanOrmawa,
        $kumpulanKegiatan,
        $kumpulanKegiatanPelatihan,
        $kumpulanLkmmPemandu,
        $kumpulanPelatihan,
        $kumpulanAbdimas,
        $kumpulanAbdimasWirausaha,
        $kumpulanAbdimasMagang,
        $kumpulanInternasionalisasi,
        $kumpulanPertukaranMhs,

        $kumpulanInternasionalisasiPelatihan,
        $kumpulanKaryaIlmiah,
        $semester
    )
    {
        $this->kumpulanKompetisi = [];
        $this->kumpulanWirausaha = [];
        $this->kumpulanMagang = [];
        $this->kumpulanOrmawa = [];
        $this->kumpulanKegiatan = [];
        $this->kumpulanKegiatanPelatihan = [];
        $this->kumpulanLkmmPemandu = [];
        $this->kumpulanPelatihan = [];
        $this->kumpulanAbdimas = [];
```

```
        $this->kumpulanAbdimasWirausaha = [];
        $this->kumpulanAbdimasMagang = [];
        $this->kumpulanInternasionalisasi = [];
        $this->kumpulanPertukaranMhs = [];

        $this->kumpulanInternasionalisasiPelatihan = [];
        $this->kumpulanKaryaIlmiah = [];

        foreach ($kumpulanKompetisi as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_kompetisi;
            $id_referensi = $aktivitas->id_kompetisi;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKompetisi, $pack);

        }
        foreach ($kumpulanWirausaha as $aktivitas)
        {
            $nama = $aktivitas->nama . " (tahun laporan " . $aktivitas-
>thn_laporan . ")";
            $id = $aktivitas->id_lapkeu_usaha;
            $id_referensi = $aktivitas->id_wirausaha;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanWirausaha, $pack);
        }
        foreach ($kumpulanMagang as $aktivitas)
        {
            $nama = $aktivitas->tempat . " (" . $aktivitas->deskripsi_kerja .
")";
            $id = $aktivitas->id_magang;
            $id_referensi = $aktivitas->id_magang;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanMagang, $pack);
        }
        foreach ($kumpulanOrmawa as $aktivitas)
```

```
        {
            $nama = $aktivitas->nama . " (" . $aktivitas->thn_mulai . "/" .
$aktivitas->thn_selesai .")";
            $id = $aktivitas->id_anggota_ormawa;
            $id_referensi = $aktivitas->id_anggota_ormawa;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanOrmawa, $pack);
        }
        foreach ($kumpulanKegiatan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_kegiatan;
            $id_referensi = $aktivitas->id_kegiatan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKegiatan, $pack);
        }
        foreach ($kumpulanKegiatanPelatihan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_pelatihan;
            $id_referensi = $aktivitas->id_pelatihan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKegiatanPelatihan, $pack);
        }
        foreach ($kumpulanLkmmPemandu as $aktivitas)
        {
            $nama = "Pemandu " . $aktivitas->nama;
            $id = $aktivitas->id_kegiatan;
            $id_referensi = $aktivitas->id_kegiatan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanLkmmPemandu, $pack);
        }
        foreach ($kumpulanPelatihan as $aktivitas)
```

```
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_pelatihan;
            $id_referensi = $aktivitas->id_pelatihan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanPelatihan, $pack);
        }
        foreach ($kumpulanAbdimas as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_abdimas;
            $id_referensi = $aktivitas->id_abdimas;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanAbdimas, $pack);
        }
        foreach ($kumpulanAbdimasWirausaha as $aktivitas)
        {
            $nama = $aktivitas->nama . " (tahun laporan " . $aktivitas-
>thn_laporan . ")";
            $id = $aktivitas->id_lapkeu_usaha;
            $id_referensi = $aktivitas->id_wirausaha;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanAbdimasWirausaha, $pack);
        }
        foreach ($kumpulanAbdimasMagang as $aktivitas)
        {
            $nama = $aktivitas->tempat . " (" . $aktivitas->deskripsi_kerja .
")";
            $id = $aktivitas->id_magang;
            $id_referensi = $aktivitas->id_magang;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanAbdimasMagang, $pack);
        }
        foreach ($kumpulanInternasionalisasi as $aktivitas)
        {
```

```
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_kegiatan;
            $id_referensi = $aktivitas->id_kegiatan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanInternasionalisasi, $pack);
        }
        foreach ($kumpulanPertukaranMhs as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_tukar_mhs;
            $id_referensi = $aktivitas->id_tukar_mhs;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanPertukaranMhs, $pack);
        }

        foreach ($kumpulanInternasionalisasiPelatihan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_pelatihan;
            $id_referensi = $aktivitas->id_pelatihan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanInternasionalisasiPelatihan, $pack);
        }

        foreach ($kumpulanKaryaIlmiah as $aktivitas)
        {
            $nama = $aktivitas->judul;
            $id = $aktivitas->id_karya_ilmiah;
            $id_referensi = $aktivitas->id_karya_ilmiah;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKaryaIlmiah, $pack);
        }
```

```
        $this->idSmt = $semester->getIdSmt()->id();
        $this->semesterKe = $semester->getSemesterKe();

    }
}
```

5.2.2.4 Mahasiswa EditSkemRequest

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\EditSkem;

class EditSkemRequest
{
    public $idMhs;
    public $idSmt;
    public $idSkem;

    public function __construct($idMhs, $idSmt, $idSkem)
    {
        $this->idMhs = $idMhs;
        $this->idSmt = $idSmt;
        $this->idSkem = $idSkem;
    }
}
```

5.2.2.5 Mahasiswa EditSkemService

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\EditSkem;

use
App\Modules\Skem\Application\Mahasiswa\EditSkem\EditSkemRequest
;
use
App\Modules\Skem\Application\Mahasiswa\EditSkem\EditSkemRespon
se;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
```

```
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\AbdimasMagangRepository;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;
use
App\Modules\Skem\Domain\Repositories\AbdimasWirausahaRepository
;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;
use
App\Modules\Skem\Domain\Repositories\KegiatanPelatihanRepository;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;
use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;
use App\Modules\Skem\Domain\Repositories\MagangRepository;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\OrmawaRepository;
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;
use App\Modules\Skem\Domain\Repositories\SemesterRepository;
use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;

use
App\Modules\Skem\Domain\Repositories\InternasionalisasiPelatihanRep
ository;

class EditSkemService
{
    private $mahasiswaRepository;
    private $semesterRepository;
    private $skemRepository;
    private $kompetisiRepository;
    private $wirausahaRepository;
    private $magangRepository;
    private $ormawaRepository;
```

```
    private $kegiatanRepository;
    private $lkmmPemanduRepository;
    private $pelatihanRepository;
    private $abdimasRepository;
    private $internasionalisasiRepository;
    private $pertukaranMhsRepository;

    private $internasionalisasiPelatihanRepository;

    public function __construct(
        MahasiswaRepository $mahasiswaRepository,
        SemesterRepository $semesterRepository,
        SkemRepository $skemRepository,
        KompetisiRepository $kompetisiRepository,
        WirausahaRepository $wirausahaRepository,
        MagangRepository $magangRepository,
        OrmawaRepository $ormawaRepository,
        KegiatanRepository $kegiatanRepository,
        KegiatanPelatihanRepository $kegiatanPelatihanRepository,
        LkmmPemanduRepository $lkmmPemanduRepository,
        AbdimasRepository $abdimasRepository,
        AbdimasWirausahaRepository $abdimasWirausahaRepository,
        AbdimasMagangRepository $abdimasMagangRepository,
        PelatihanRepository $pelatihanRepository,
        InternasionalisasiRepository $internasionalisasiRepository,
        PertukaranMhsRepository $pertukaranMhsRepository,

        InternasionalisasiPelatihanRepository
$internasionalisasiPelatihanRepository
    )
    {
        $this->mahasiswaRepository = $mahasiswaRepository;
        $this->semesterRepository = $semesterRepository;
        $this->skemRepository = $skemRepository;
        $this->kompetisiRepository = $kompetisiRepository;
        $this->wirausahaRepository = $wirausahaRepository;
        $this->magangRepository = $magangRepository;
        $this->ormawaRepository = $ormawaRepository;
        $this->kegiatanRepository = $kegiatanRepository;
```

```
        $this->kegiatanPelatihanRepository                                    =
$kegiatanPelatihanRepository;
        $this->lkmmPemanduRepository = $lkmmPemanduRepository;
        $this->abdimasRepository = $abdimasRepository;
        $this->abdimasWirausahaRepository                                     =
$abdimasWirausahaRepository;
        $this->abdimasMagangRepository = $abdimasMagangRepository;
        $this->pelatihanRepository = $pelatihanRepository;
        $this->internasionalisasiRepository = $internasionalisasiRepository;
        $this->pertukaranMhsRepository = $pertukaranMhsRepository;

        $this->internasionalisasiPelatihanRepository                          =
$internasionalisasiPelatihanRepository;
    }

    public function handle(EditSkemRequest $editAjuanSkemRequest)
    {
        $idMhs = new MahasiswaId($editAjuanSkemRequest->idMhs) ;
        $idSmt = new SemesterId($editAjuanSkemRequest->idSmt);
        $idSkem = new SkemId($editAjuanSkemRequest->idSkem);
        $mahasiswa = $this->mahasiswaRepository->get($idMhs);
        $semester = $mahasiswa->getSemester($idSmt);
        $periode = $mahasiswa->getThnAngkatan() == 2018 ? 21557 : 366;

        $kumpulanIdKegiatanSkem          =          $this->skemRepository-
>getSemuaIdPortofolioKegiatanSkemForEdit($idSkem);

        $kumpulanKompetisi          =          $this->kompetisiRepository-
>requestListForUpdate($idMhs, $periode, $idSkem);

        $kumpulanWirausaha          =          $this->wirausahaRepository-
>requestListForUpdate($idMhs, $periode, $idSkem);
//        $kumpulanWirausaha = [];
        $kumpulanMagang          =          $this->magangRepository-
>requestListForUpdate($idMhs, $periode, $idSkem);
//        $kumpulanMagang = [];
```

```
        $kumpulanOrmawa = $this->ormawaRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanKegiatan = $this->kegiatanRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanKegiatanPelatihan = $this->kegiatanPelatihanRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanLkmmPemandu = $this->lkmmPemanduRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanAbdimas = $this->abdimasRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanAbdimasWirausaha = $this->abdimasWirausahaRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanAbdimasMagang = $this->abdimasMagangRepository->requestListForUpdate($idMhs, $periode, $idSkem);

        $kumpulanPelatihan = $this->pelatihanRepository->requestListForUpdate($idMhs, $periode, $idSkem);

        $kumpulanInternasionalisasi = $this->internasionalisasiRepository->requestListForUpdate($idMhs, $periode, $idSkem);
        $kumpulanPertukaranMhs = $this->pertukaranMhsRepository->requestListForUpdate($idMhs, $periode, $idSkem);

        $kumpulanInternasionalisasiPelatihan = $this->internasionalisasiPelatihanRepository->requestListForUpdate($idMhs, $periode, $idSkem);

        $response = new EditSkemResponse(
            $kumpulanKompetisi,
            $kumpulanWirausaha,
            $kumpulanMagang,
            $kumpulanOrmawa,
            $kumpulanKegiatan,
            $kumpulanKegiatanPelatihan,
            $kumpulanLkmmPemandu,
            $kumpulanPelatihan,
            $kumpulanAbdimas,
            $kumpulanAbdimasWirausaha,
```

```
            $kumpulanAbdimasMagang,
            $kumpulanInternasionalisasi,
            $kumpulanPertukaranMhs,

            $kumpulanInternasionalisasiPelatihan,

            $semester,
            $kumpulanIdKegiatanSkem,
            $idSkem
        );
        return $response;
    }
}
```

5.2.2.6 Mahasiswa EditSkemResponse

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\EditSkem;

class EditSkemResponse
{
    public $kumpulanKompetisi;
    public $kumpulanWirausaha;
    public $kumpulanMagang;
    public $kumpulanOrmawa;
    public $kumpulanKegiatan;
    public $kumpulanKegiatanPelatihan;
    public $kumpulanLkmmPemandu;
    public $kumpulanPelatihan;
    public $kumpulanAbdimas;
    public $kumpulanAbdimasWirausaha;
    public $kumpulanAbdimasMagang;
    public $kumpulanInternasionalisasi;
    public $kumpulanPertukaranMhs;

    public $kumpulanInternasionalisasiPelatihan;
```

```
    public $idSmt;
    public $semesterKe;
    public $kumpulanIdKegiatanSkem;
    public $idSkem;

    public function __construct(
        $kumpulanKompetisi,
        $kumpulanWirausaha,
        $kumpulanMagang,
        $kumpulanOrmawa,
        $kumpulanKegiatan,
        $kumpulanKegiatanPelatihan,
        $kumpulanLkmmPemandu,
        $kumpulanPelatihan,
        $kumpulanAbdimas,
        $kumpulanAbdimasWirausaha,
        $kumpulanAbdimasMagang,
        $kumpulanInternasionalisasi,
        $kumpulanPertukaranMhs,

        $kumpulanInternasionalisasiPelatihan,

        $semester,
        $kumpulanIdKegiatanSkem,
        $idSkem
    )
    {
        $this->kumpulanKompetisi = [];
        $this->kumpulanWirausaha = [];
        $this->kumpulanMagang = [];
        $this->kumpulanOrmawa = [];
        $this->kumpulanKegiatan = [];
        $this->kumpulanKegiatanPelatihan = [];
        $this->kumpulanLkmmPemandu = [];
        $this->kumpulanPelatihan = [];
        $this->kumpulanAbdimas = [];
        $this->kumpulanAbdimasWirausaha = [];
        $this->kumpulanAbdimasMagang = [];
```

```
        $this->kumpulanInternasionalisasi = [];
        $this->kumpulanPertukaranMhs = [];

        $this->kumpulanInternasionalisasiPelatihan = [];

        foreach ($kumpulanKompetisi as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_kompetisi;
            $id_referensi = $aktivitas->id_kompetisi;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKompetisi, $pack);

        }

        foreach ($kumpulanWirausaha as $aktivitas)
        {
            $nama = $aktivitas->nama . " (tahun laporan " . $aktivitas-
>thn_laporan . ")";
            $id = $aktivitas->id_lapkeu_usaha;
            $id_referensi = $aktivitas->id_wirausaha;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanWirausaha, $pack);
        }

        foreach ($kumpulanMagang as $aktivitas)
        {
            $nama = $aktivitas->tempat . " (" . $aktivitas->deskripsi_kerja .
")";
            $id = $aktivitas->id_magang;
            $id_referensi = $aktivitas->id_magang;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanMagang, $pack);
        }
```

```
        foreach ($kumpulanOrmawa as $aktivitas)
        {
            $nama = $aktivitas->nama . " (" . $aktivitas->thn_mulai . "/" .
$aktivitas->thn_selesai .")";
            $id = $aktivitas->id_anggota_ormawa;
            $id_referensi = $aktivitas->id_anggota_ormawa;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanOrmawa, $pack);
        }

        foreach ($kumpulanKegiatan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_kegiatan;
            $id_referensi = $aktivitas->id_kegiatan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKegiatan, $pack);
        }

        foreach ($kumpulanKegiatanPelatihan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_pelatihan;
            $id_referensi = $aktivitas->id_pelatihan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanKegiatanPelatihan, $pack);
        }

        foreach ($kumpulanLkmmPemandu as $aktivitas)
        {
            $nama = "Pemandu " . $aktivitas->nama;
            $id = $aktivitas->id_kegiatan;
            $id_referensi = $aktivitas->id_kegiatan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
```

```
            array_push($this->kumpulanLkmmPemandu, $pack);
        }

        foreach ($kumpulanPelatihan as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_pelatihan;
            $id_referensi = $aktivitas->id_pelatihan;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanPelatihan, $pack);
        }

        foreach ($kumpulanAbdimas as $aktivitas)
        {
            $nama = $aktivitas->nama;
            $id = $aktivitas->id_abdimas;
            $id_referensi = $aktivitas->id_abdimas;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanAbdimas, $pack);
        }

        foreach ($kumpulanAbdimasWirausaha as $aktivitas)
        {
            $nama = $aktivitas->nama . " (tahun laporan " . $aktivitas-
>thn_laporan . ")";
            $id = $aktivitas->id_lapkeu_usaha;
            $id_referensi = $aktivitas->id_wirausaha;
            $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
            array_push($this->kumpulanAbdimasWirausaha, $pack);
        }

        foreach ($kumpulanAbdimasMagang as $aktivitas)
        {
```

```
        $nama = $aktivitas->tempat . " (" . $aktivitas->deskripsi_kerja .
")";
        $id = $aktivitas->id_magang;
        $id_referensi = $aktivitas->id_magang;
        $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
        array_push($this->kumpulanAbdimasMagang, $pack);
      }

      foreach ($kumpulanInternasionalisasi as $aktivitas)
      {
        $nama = $aktivitas->nama;
        $id = $aktivitas->id_kegiatan;
        $id_referensi = $aktivitas->id_kegiatan;
        $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
        array_push($this->kumpulanInternasionalisasi, $pack);
      }

      foreach ($kumpulanPertukaranMhs as $aktivitas)
      {
        $nama = $aktivitas->nama;
        $id = $aktivitas->id_tukar_mhs;
        $id_referensi = $aktivitas->id_tukar_mhs;
        $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
        array_push($this->kumpulanPertukaranMhs, $pack);
      }

      foreach ($kumpulanInternasionalisasiPelatihan as $aktivitas)
      {
        $nama = $aktivitas->nama;
        $id = $aktivitas->id_pelatihan;
        $id_referensi = $aktivitas->id_pelatihan;
        $pack = ['nama' => $nama, 'id' => $id, "id_referensi" =>
$id_referensi];
        array_push($this->kumpulanInternasionalisasiPelatihan, $pack);
      }
```

```
        $this->idSmt = $semester->getIdSmt()->id();
        $this->semesterKe = $semester->getSemesterKe();

        $this->idSkem = $idSkem->id();
        $this->kumpulanIdKegiatanSkem = $kumpulanIdKegiatanSkem;

    }
}
```

5.2.2.7 Mahasiswa IndexSkemService

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\IndexSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\SkemRepository;

use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Mahasiswa\Mahasiswa;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;

use
App\Modules\Skem\Application\Mahasiswa\IndexSkem\IndexSkemResp
onse;

class IndexSkemService
{
    private $skemRepository;
    private $mahasiswaRepository;

    public function __construct(MahasiswaRepository
$mahasiswaRepository, SkemRepository $skemRepository)
    {
        $this->skemRepository = $skemRepository;
```

```
        $this->mahasiswaRepository = $mahasiswaRepository;
    }
    public function handle($request)
    {
        $idMhs = new MahasiswaId($request);
        $mahasiswa = $this->mahasiswaRepository->get($idMhs);
        $kumpulanSkem = $this->skemRepository-
>getSemuaSkemView($idMhs);
        $mahasiswa->addKumpulanSkem($kumpulanSkem);


        $response = new IndexSkemResponse($mahasiswa);
        return $response;
    }
}
```

5.2.2.8 Mahasiswa IndexSkemResponse

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\IndexSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa\Mahasiswa;
use App\Modules\Skem\Domain\Model\Semester\Semester;

class IndexSkemResponse
{
    public $kumpulanSemester;

    public function __construct(Mahasiswa $mahasiswa)
    {

        foreach ($mahasiswa->getSemuaSemester() as $semester) {

            $idSmt = $semester->getIdSmt()->id();
            $namaSemester = $semester->getNama();
            $semesterKe = $semester->getSemesterKe();
            $nama = $semester->getNama();
```

```
            $skem = $semester->getSkem();
            if ($skem != null) {
                $idSkem = $skem->getIdSkem()->id();
                $statusAjuan = $skem->getStatusAjuan();
                $statusLulus = $skem->getStatusLulus();
                $totalSkemA = $skem->getTotalProdukA();
                $totalSkemB = $skem->getTotalProdukB();
                $totalSkemC = $skem->GetTotalProdukC();
                $totalSkemD = $skem->getTotalProdukD();
                $totalNilaiSkem = $skem->getTotalProdukSkem();
                $ipSkem = $skem->getIpSkem();
                $skemDTO = [
                    'idSkem' => $idSkem,
                    'statusAjuan' => $statusAjuan,
                    'statusLulus' => $statusLulus,
                    'totalSkemA' => $totalSkemA,
                    'totalSkemB' => $totalSkemB,
                    'totalSkemC' => $totalSkemC,
                    'totalSkemD' => $totalSkemD,
                    'totalNilaiSkem' => $totalNilaiSkem,
                    'ipSkem' => $ipSkem
                ];
            } else {
                $skemDTO = null;
            }

            $semesterDTO = [
                'idSmt' => $idSmt,
                'namaSemester' => $namaSemester,
                'semesterKe' => $semesterKe,
                'nama' => $nama,
                'skem' => $skemDTO
            ];

            $this->kumpulanSemester[] = $semesterDTO;
        }
    }
```

```
}
```

5.2.2.9 Mahasiswa ShowSkemService

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\ShowSkem;

use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\SkemRepository;

use App\Modules\Skem\Domain\Repositories\SemesterRepository;

use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;
use App\Modules\Skem\Domain\Repositories\MagangRepository;

use App\Modules\Skem\Domain\Repositories\OrmawaRepository;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;

use App\Modules\Skem\Domain\Repositories\AbdimasRepository;

use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;

use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;

class ShowSkemService
{
   private $skemRepository;

   public function __construct(SkemRepository $skemRepository)
   {
      $this->skemRepository = $skemRepository;
```

```
    }

    public function handle($request)
    {
        $idSkem = new SkemId($request);

        $skemDenganKumpulanKegiatanSkem = $this->skemRepository-
>getSkemDenganKumpulanKegiatanSkem($idSkem);
//        dd($skemDenganKumpulanKegiatanSkem);
        $response = new ShowSkemResponse(
            $skemDenganKumpulanKegiatanSkem
        );
//        dd($response);
        return $response;


    }
}
```

5.2.2.10 Mahasiswa ShowSkemResponse

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\ShowSkem;

use App\Modules\Skem\Domain\Model\Skem\Skem;

class ShowSkemResponse
{
    public $idSkem;
    public $idSmt;
    public $ipSkem;
    public $semesterKe;
    public $statusLulus;
    public $statusAjuan;
    public $komentar;
    public $kumpulanAktivitasSkemA;
```

```
    public $kumpulanAktivitasSkemB;
    public $kumpulanAktivitasSkemC;
    public $kumpulanAktivitasSkemD;

    public function __construct(Skem $skem)
    {
        $kumpulanAktivitasSkemA = [];
        $kumpulanAktivitasSkemB = [];
        $kumpulanAktivitasSkemC = [];
        $kumpulanAktivitasSkemD = [];

//      $kumpulanKegiatanSkem = $kumpulanKegiatanSkem;
        foreach ($skem->getKumpulanAktivitas() as $kegiatanSkem) {
            $aktivitas = ['namaKegiatan' => $kegiatanSkem-
>getNamaKegiatan(),
                'idKegiatanSkem' => $kegiatanSkem->getIdKegiatanSkem()-
>id(),
                'idBidangSkem' => $kegiatanSkem->getIdBidangSkem(),
                'idPortofolio' => $kegiatanSkem->getIdPortofolio(),
                'kreditSkem' => $kegiatanSkem->getKreditSkem(),
                'nilaiAngka' => $kegiatanSkem->getNilaiAngka()
            ];
            if(in_array($kegiatanSkem->getIdBidangSkem(), [2,3,4])) {
                array_push($kumpulanAktivitasSkemA, $aktivitas);
            } elseif (in_array($kegiatanSkem->getIdBidangSkem(), [6,7,10]))
{
                array_push($kumpulanAktivitasSkemB, $aktivitas);
            } elseif (in_array($kegiatanSkem->getIdBidangSkem(), [8])) {
                array_push($kumpulanAktivitasSkemC, $aktivitas);
            } else {
                array_push($kumpulanAktivitasSkemD, $aktivitas);
            }
        }
        $this->kumpulanAktivitasSkemA = $kumpulanAktivitasSkemA;
        $this->kumpulanAktivitasSkemB = $kumpulanAktivitasSkemB;
        $this->kumpulanAktivitasSkemC = $kumpulanAktivitasSkemC;
        $this->kumpulanAktivitasSkemD = $kumpulanAktivitasSkemD;

        $this->idSkem = $skem->getIdSkem()->id();
```

```
        $this->idSmt = $skem->getIdSmt()->id();
        $this->ipSkem = $skem->getIpSkem();
        $this->semesterKe = $skem->getSemesterKe();
        $this->statusLulus = $skem->getStatusLulus();
        $this->statusAjuan = $skem->getStatusAjuan();
        $this->komentar = $skem->getKomentar();
    }
}
```

5.2.2.11 Mahasiswa ShowKegiatanSkemService

```
<?php

namespace
App\Modules\Skem\Application\Mahasiswa\ShowKegiatanSkem;

use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;

class ShowKegiatanSkemService
{
    private $skemRepository;

    public function __construct(
        SkemRepository $skemRepository
    )
    {
        $this->skemRepository = $skemRepository;
    }

    public function handle($idKegiatanSkem)
    {
        $kegiatanSkem = $this->skemRepository-
>getKegiatanSkemDenganKumpulanElemenPenilaian(new
AktivitasId($idKegiatanSkem));
        $response = new ShowKegiatanSkemResponse($kegiatanSkem);
//        dd($response);
```

```
        return $response;


    }
}
```

5.2.2.12 Mahasiswa ShowKegiatanSkemResponse

```
<?php

namespace
App\Modules\Skem\Application\Mahasiswa\ShowKegiatanSkem;

class ShowKegiatanSkemResponse
{
    public $namaKegiatan;
    public $kreditSkem;
    public $nilaiAngka;
    public $nilaiHuruf;
    public $produkSkem;
    public $tglKlaim;
    public $idJenisPortofolio;
    public $idSkem;
    public $kumpulanElemenPenilaian;

    public function __construct($kegiatanSkem)
    {
        $jenisPortofolio = $kegiatanSkem->getIdJenisPortofolio() ==
'pertukaran_mhs' ? 'pertukaran mahasiswa' : $kegiatanSkem-
>getIdJenisPortofolio();

        $this->namaKegiatan = $kegiatanSkem->getNamaKegiatan();
        $this->kreditSkem = $kegiatanSkem->getKreditSkem();
        $this->nilaiAngka = $kegiatanSkem->getNilaiAngka();
        $this->nilaiHuruf = $kegiatanSkem->getNilaiHuruf();
        $this->produkSkem = $kegiatanSkem->getProdukSkem();
        $this->tglKlaim = $kegiatanSkem->getTglKlaim();
        $this->idJenisPortofolio = $jenisPortofolio;
        $this->idSkem = $kegiatanSkem->getIdSkem()->id();
```

```
        $kumpulanElemenPenilaianDTO = [];
        foreach ($kegiatanSkem->getKumpulanElemenPenilaian() as
$elemenPenilaian)
        {
            $elemenPenilaianDTO = [
                'nama' => $elemenPenilaian->getNama(),
                'value' => $elemenPenilaian->getValue(),
                'bobot' => $elemenPenilaian->getBobot()
            ];
            array_push($kumpulanElemenPenilaianDTO,
$elemenPenilaianDTO);
        }

        $this->kumpulanElemenPenilaian =
$kumpulanElemenPenilaianDTO;
    }
}
```

5.2.2.13 Mahasiswa StoreSkemRequest

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\StoreSkem;

class StoreSkemRequest
{
    public $idSmt;
    public $idMhs;
    public $kompetisi;
    public $wirausaha;
    public $magang;
    public $ormawa;
    public $kegiatan;
    public $kegiatanPelatihan;
    public $lkmmPemandu;
```

```
    public $pelatihan;
    public $abdimas;
    public $abdimasWirausaha;
    public $abdimasMagang;
    public $internasionalisasi;
    public $pertukaranMhs;

    public $internasionalisasiPelatihan;

    public function __construct(
        $idSmt,
        $idMhs,
        $kompetisi,
        $wirausaha,
        $magang,
        $ormawa,
        $kegiatan,
        $kegiatanPelatihan,
        $lkmmPemandu,
        $pelatihan,
        $abdimas,
        $abdimasWirausaha,
        $abdimasMagang,
        $internasionalisasi,
        $pertukaranMhs,

        $internasionalisasiPelatihan
    )
    {
        $this->idSmt = $idSmt;
        $this->idMhs = $idMhs;
        $this->kompetisi = $kompetisi;
        $this->wirausaha = $wirausaha;
        $this->magang = $magang;
        $this->ormawa = $ormawa;
        $this->kegiatan = $kegiatan;
        $this->kegiatanPelatihan = $kegiatanPelatihan;
        $this->lkmmPemandu = $lkmmPemandu;
        $this->pelatihan = $pelatihan;
```

```
        $this->abdimas = $abdimas;
        $this->abdimasWirausaha = $abdimasWirausaha;
        $this->abdimasMagang = $abdimasMagang;
        $this->internasionalisasi = $internasionalisasi;
        $this->pertukaranMhs = $pertukaranMhs;

        $this->internasionalisasiPelatihan = $internasionalisasiPelatihan;
    }
}
```

5.2.2.14 Mahasiswa StoreSkemService

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\StoreSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;



use
App\Modules\Skem\Domain\Repositories\AbdimasMagangRepository;
use
App\Modules\Skem\Domain\Repositories\AbdimasWirausahaRepository
;
use
App\Modules\Skem\Domain\Repositories\KegiatanPelatihanRepository;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\SemesterRepository;
use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;
use App\Modules\Skem\Domain\Repositories\MagangRepository;
use App\Modules\Skem\Domain\Repositories\OrmawaRepository;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;
```

```
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;

use
App\Modules\Skem\Domain\Repositories\InternasionalisasiPelatihanRep
ository;

use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class StoreSkemService
{
    private $mahasiswaRepository;
    private $semesterRepository;
    private $skemRepository;
    private $kompetisiRepository;
    private $wirausahaRepository;
    private $magangRepository;
    private $ormawaRepository;
    private $kegiatanRepository;
    private $kegiatanPelatihanRepository;
    private $lkmmPemanduRepository;
    private $abdimasRepository;
    private $abdimasWirausahaRepository;
    private $abdimasMagangRepository;
    private $pelatihanRepository;
    private $internasionalisasiRepository;
    private $pertukaranMhsRepository;

    private $internasionalisasiPelatihanRepository;

    public function __construct(
        MahasiswaRepository $mahasiswaRepository,
        SemesterRepository $semesterRepository,
        SkemRepository $skemRepository,
        KompetisiRepository $kompetisiRepository,
```

```
        WirausahaRepository $wirausahaRepository,
        MagangRepository $magangRepository,
        OrmawaRepository $ormawaRepository,
        KegiatanRepository $kegiatanRepository,
        KegiatanPelatihanRepository $kegiatanPelatihanRepository,
        LkmmPemanduRepository $lkmmPemanduRepository,
        AbdimasRepository $abdimasRepository,
        AbdimasWirausahaRepository $abdimasWirausahaRepository,
        AbdimasMagangRepository $abdimasMagangRepository,
        PelatihanRepository $pelatihanRepository,
        InternasionalisasiRepository $internasionalisasiRepository,
        PertukaranMhsRepository $pertukaranMhsRepository,

        InternasionalisasiPelatihanRepository
$internasionalisasiPelatihanRepository
    )
    {
        $this->mahasiswaRepository = $mahasiswaRepository;
        $this->semesterRepository = $semesterRepository;
        $this->skemRepository = $skemRepository;
        $this->kompetisiRepository = $kompetisiRepository;
        $this->wirausahaRepository = $wirausahaRepository;
        $this->magangRepository = $magangRepository;
        $this->ormawaRepository = $ormawaRepository;
        $this->kegiatanRepository = $kegiatanRepository;
        $this->kegiatanPelatihanRepository                    =
$kegiatanPelatihanRepository;
        $this->lkmmPemanduRepository = $lkmmPemanduRepository;
        $this->abdimasRepository = $abdimasRepository;
        $this->abdimasWirausahaRepository                     =
$abdimasWirausahaRepository;
        $this->abdimasMagangRepository = $abdimasMagangRepository;
        $this->pelatihanRepository = $pelatihanRepository;
        $this->internasionalisasiRepository = $internasionalisasiRepository;
        $this->pertukaranMhsRepository = $pertukaranMhsRepository;
```

```
        $this->internasionalisasiPelatihanRepository                    =
$internasionalisasiPelatihanRepository;
    }

    public function handle($request)
    {
        $idMhs = new MahasiswaId($request->idMhs);
        $idSmt = new SemesterId($request->idSmt);
        $mahasiswa = $this->mahasiswaRepository->get($idMhs);
        $semester = $mahasiswa->getSemester($idSmt);
        $periode = $mahasiswa->getThnAngkatan() == 2018 ? 21557 : 366;

        $idSkem = new SkemId();
        $skem = Skem::make(
            $idSkem,
            $idMhs,
            $idSmt,
            $semester->getSemesterKe()
        );

        $kumpulanKompetisi = (empty($request->kompetisi))? null : $this-
>kompetisiRepository->buildKumpulanKompetisi($request->kompetisi,
$idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanKompetisi != null) {
            $skem->addKumpulanAktivitas($kumpulanKompetisi);
        }
//        dd($kumpulanKompetisi);

        $kumpulanWirausaha = (empty($request->wirausaha))? null : $this-
>wirausahaRepository->buildKumpulanWirausaha($request->wirausaha,
$idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanWirausaha != null) {
            $skem->addKumpulanAktivitas($kumpulanWirausaha);
        }
//        dd($kumpulanWirausaha);

        $kumpulanMagang = (empty($request->magang))? null : $this-
>magangRepository->buildKumpulanMagang($request->magang,
$idMhs, $idSkem, date("Y-m-d"), $periode);
```

```
        if ($kumpulanMagang != null) {
            $skem->addKumpulanAktivitas($kumpulanMagang);
        }

        $kumpulanOrmawa = (empty($request->ormawa))? null : $this->ormawaRepository->buildKumpulanOrmawa($request->ormawa, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanOrmawa != null) {
            $skem->addKumpulanAktivitas($kumpulanOrmawa);
        }

        $kumpulanKegiatan = (empty($request->kegiatan))? null : $this->kegiatanRepository->buildKumpulanKegiatan($request->kegiatan, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanKegiatan != null) {
            $skem->addKumpulanAktivitas($kumpulanKegiatan);
        }

        $kumpulanKegiatanPelatihan = (empty($request->kegiatanPelatihan))? null : $this->kegiatanPelatihanRepository->buildKumpulanKegiatan($request->kegiatanPelatihan, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanKegiatanPelatihan != null) {
            $skem->addKumpulanAktivitas($kumpulanKegiatanPelatihan);
        }

        $kumpulanLkmmPemandu = (empty($request->lkmmPemandu))? null : $this->lkmmPemanduRepository->buildKumpulanLkmmPemandu($request->lkmmPemandu, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanLkmmPemandu != null) {
            $skem->addKumpulanAktivitas($kumpulanLkmmPemandu);
        }

        $kumpulanPelatihan = (empty($request->pelatihan))? null : $this->pelatihanRepository->buildKumpulanPelatihan($request->pelatihan, $idMhs, $idSkem, date("Y-m-d"), $periode);
```

```
        if ($kumpulanPelatihan != null) {
            $skem->addKumpulanAktivitas($kumpulanPelatihan);
        }

        $kumpulanAbdimas = (empty($request->abdimas))? null : $this->abdimasRepository->buildKumpulanAbdimas($request->abdimas, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimas != null) {
            $skem->addKumpulanAktivitas($kumpulanAbdimas);
        }

        $kumpulanAbdimasWirausaha = (empty($request->abdimasWirausaha))? null : $this->abdimasWirausahaRepository->buildKumpulanAbdimas($request->abdimasWirausaha, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimasWirausaha != null) {
            $skem->addKumpulanAktivitas($kumpulanAbdimasWirausaha);
        }

        $kumpulanAbdimasMagang = (empty($request->abdimasMagang))? null : $this->abdimasMagangRepository->buildKumpulanAbdimas($request->abdimasMagang, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimasMagang != null) {
            $skem->addKumpulanAktivitas($kumpulanAbdimasMagang);
        }

        $kumpulanInternasionalisasi = (empty($request->internasionalisasi))? null : $this->internasionalisasiRepository->buildKumpulanInternasionalisasi($request->internasionalisasi, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanInternasionalisasi != null) {
            $skem->addKumpulanAktivitas($kumpulanInternasionalisasi);
        }

        $kumpulanPertukaranMhs = (empty($request->pertukaranMhs))? null : $this->pertukaranMhsRepository->buildKumpulanPertukaranMhs($request->pertukaranMhs, $idMhs, $idSkem, date("Y-m-d"), $periode);
```

```
        if ($kumpulanPertukaranMhs != null) {
            $skem->addKumpulanAktivitas($kumpulanPertukaranMhs);
        }

        $kumpulanInternasionalisasiPelatihan = (empty($request->internasionalisasiPelatihan))? null : $this->internasionalisasiPelatihanRepository->buildKumpulanInternasionalisasi($request->internasionalisasiPelatihan, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanInternasionalisasiPelatihan != null) {
            $skem->addKumpulanAktivitas($kumpulanInternasionalisasiPelatihan);
        }

//        dd($skem);

        $success = $this->skemRepository->add($skem);
        $response = $skem->getIdSkem()->id();
        return $response;

    }
}
```

5.2.2.15 Mahasiswa UpdateSkemRequest

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\UpdateSkem;

class UpdateSkemRequest
{
    public $idMhs;
    public $idSkem;
    public $kompetisi;
    public $wirausaha;
    public $magang;
    public $ormawa;
```

```
    public $kegiatan;
    public $kegiatanPelatihan;
    public $lkmmPemandu;
    public $pelatihan;
    public $abdimas;
    public $abdimasWirausaha;
    public $abdimasMagang;
    public $internasionalisasi;
    public $pertukaranMhs;

    public $internasionalisasiPelatihan;

    public function __construct(

        $idMhs,
        $idSkem,
        $kompetisi,
        $wirausaha,
        $magang,
        $ormawa,
        $kegiatan,
        $kegiatanPelatihan,
        $lkmmPemandu,
        $pelatihan,
        $abdimas,
        $abdimasWirausaha,
        $abdimasMagang,
        $internasionalisasi,
        $pertukaranMhs,

        $internasionalisasiPelatihan
    )
    {
        $this->idMhs = $idMhs;
        $this->idSkem = $idSkem;
        $this->kompetisi = $kompetisi;
        $this->wirausaha = $wirausaha;
        $this->magang = $magang;
        $this->ormawa = $ormawa;
```

```
        $this->kegiatan = $kegiatan;
        $this->kegiatanPelatihan = $kegiatanPelatihan;
        $this->lkmmPemandu = $lkmmPemandu;
        $this->pelatihan = $pelatihan;
        $this->abdimas = $abdimas;
        $this->abdimasWirausaha = $abdimasWirausaha;
        $this->abdimasMagang = $abdimasMagang;
        $this->internasionalisasi = $internasionalisasi;
        $this->pertukaranMhs = $pertukaranMhs;

        $this->internasionalisasiPelatihan = $internasionalisasiPelatihan;

        $kumpulanIdInput = [];

        if($kompetisi != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$kompetisi);
        }

        if($wirausaha != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$wirausaha);
        }

        if($magang != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput, $magang);
        }

        if($ormawa != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput, $ormawa);
        }

        if($kegiatan != null)
```

```
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput, $kegiatan);
        }

        if($kegiatanPelatihan != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$kegiatanPelatihan);
        }

        if($lkmmPemandu != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$lkmmPemandu);
        }

        if($pelatihan != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput, $pelatihan);
        }

        if($abdimas != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput, $abdimas);
        }

        if($abdimasWirausaha != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$abdimasWirausaha);
        }

        if($abdimasMagang != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$abdimasMagang);
        }

        if($internasionalisasi != null)
```

```
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$internasionalisasi);
        }

        if($pertukaranMhs != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$pertukaranMhs);
        }

        if($internasionalisasiPelatihan != null)
        {
            $kumpulanIdInput = array_merge($kumpulanIdInput,
$internasionalisasiPelatihan);
        }

        $this->kumpulanIdInput = $kumpulanIdInput;
    }
}
```

5.2.2.16 Mahasiswa UpdateSkemService

```
<?php

namespace App\Modules\Skem\Application\Mahasiswa\UpdateSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa\Mahasiswa;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\AbdimasMagangRepository;
use
App\Modules\Skem\Domain\Repositories\AbdimasWirausahaRepository
;
```

```
use
App\Modules\Skem\Domain\Repositories\KegiatanPelatihanRepository;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Repositories\SemesterRepository;
use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;
use App\Modules\Skem\Domain\Repositories\MagangRepository;
use App\Modules\Skem\Domain\Repositories\OrmawaRepository;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;

use
App\Modules\Skem\Domain\Repositories\InternasionalisasiPelatihanRep
ository;

use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;

class UpdateSkemService
{
    private $mahasiswaRepository;
    private $semesterRepository;
    private $skemRepository;
    private $kompetisiRepository;
    private $wirausahaRepository;
    private $magangRepository;
    private $ormawaRepository;
    private $kegiatanRepository;
    private $kegiatanPelatihanRepository;
    private $lkmmPemanduRepository;
    private $abdimasRepository;
    private $abdimasWirausahaRepository;
```

```
    private $abdimasMagangRepository;
    private $pelatihanRepository;
    private $internasionalisasiRepository;
    private $pertukaranMhsRepository;

    private $internasionalisasiPelatihanRepository;

    public function __construct(
        MahasiswaRepository $mahasiswaRepository,
        SemesterRepository $semesterRepository,
        SkemRepository $skemRepository,
        KompetisiRepository $kompetisiRepository,
        WirausahaRepository $wirausahaRepository,
        MagangRepository $magangRepository,
        OrmawaRepository $ormawaRepository,
        KegiatanRepository $kegiatanRepository,
        KegiatanPelatihanRepository $kegiatanPelatihanRepository,
        LkmmPemanduRepository $lkmmPemanduRepository,
        AbdimasRepository $abdimasRepository,
        AbdimasWirausahaRepository $abdimasWirausahaRepository,
        AbdimasMagangRepository $abdimasMagangRepository,
        PelatihanRepository $pelatihanRepository,
        InternasionalisasiRepository $internasionalisasiRepository,
        PertukaranMhsRepository $pertukaranMhsRepository,

        InternasionalisasiPelatihanRepository
$internasionalisasiPelatihanRepository
    )
    {
        $this->mahasiswaRepository = $mahasiswaRepository;
        $this->semesterRepository = $semesterRepository;
        $this->skemRepository = $skemRepository;
        $this->kompetisiRepository = $kompetisiRepository;
        $this->wirausahaRepository = $wirausahaRepository;
        $this->magangRepository = $magangRepository;
        $this->ormawaRepository = $ormawaRepository;
        $this->kegiatanRepository = $kegiatanRepository;
```

```
        $this->kegiatanPelatihanRepository                                  =
$kegiatanPelatihanRepository;
        $this->lkmmPemanduRepository = $lkmmPemanduRepository;
        $this->abdimasRepository = $abdimasRepository;
        $this->abdimasWirausahaRepository                                   =
$abdimasWirausahaRepository;
        $this->abdimasMagangRepository = $abdimasMagangRepository;
        $this->pelatihanRepository = $pelatihanRepository;
        $this->internasionalisasiRepository = $internasionalisasiRepository;
        $this->pertukaranMhsRepository = $pertukaranMhsRepository;

        $this->internasionalisasiPelatihanRepository                        =
$internasionalisasiPelatihanRepository;
    }
    public function handle($request)
    {

        $idMhs = new MahasiswaId($request->idMhs);
        $idSkem = new SkemId($request->idSkem);
        $skem = $this->skemRepository->getSkem($idSkem);
        $idSmt = $skem->getIdSmt();

        $mahasiswa = $this->mahasiswaRepository->get($idMhs);
        $semester = $mahasiswa->getSemester($idSmt);
        $periode = $mahasiswa->getThnAngkatan() == 2018 ? 21557 : 366;

        $skemBaru = Skem::make(
            $idSkem,
            $idMhs,
            $idSmt,
            $skem->getSemesterKe()
        );

        $KumpulanKompetisi = (empty($request->kompetisi))? null : $this-
>kompetisiRepository->updateKumpulanKompetisi($request-
>kompetisi, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($KumpulanKompetisi != null) {
            $skemBaru->addKumpulanAktivitas($KumpulanKompetisi);
        }
```

```
    $kumpulanWirausaha = (empty($request->wirausaha))? null : $this->wirausahaRepository->updateKumpulanWirausaha($request->wirausaha, $idMhs, $idSkem, date("Y-m-d"), $periode);
    if ($kumpulanWirausaha != null) {
        $skemBaru->addKumpulanAktivitas($kumpulanWirausaha);
    }

    $kumpulanMagang = (empty($request->magang))? null : $this->magangRepository->updateKumpulanMagang($request->magang, $idMhs, $idSkem, date("Y-m-d"), $periode);
    if ($kumpulanMagang != null) {
        $skemBaru->addKumpulanAktivitas($kumpulanMagang);
    }

    $kumpulanOrmawa = (empty($request->ormawa))? null : $this->ormawaRepository->updateKumpulanOrmawa($request->ormawa, $idMhs, $idSkem, date("Y-m-d"), $periode);
    if ($kumpulanOrmawa != null) {
        $skemBaru->addKumpulanAktivitas($kumpulanOrmawa);
    }

    $kumpulanKegiatan = (empty($request->kegiatan))? null : $this->kegiatanRepository->updateKumpulankegiatan($request->kegiatan, $idMhs, $idSkem, date("Y-m-d"), $periode);
    if ($kumpulanKegiatan != null) {
        $skemBaru->addKumpulanAktivitas($kumpulanKegiatan);
    }

    $kumpulanKegiatanPelatihan = (empty($request->kegiatanPelatihan))? null : $this->kegiatanPelatihanRepository->updateKumpulankegiatan($request->kegiatanPelatihan, $idMhs, $idSkem, date("Y-m-d"), $periode);
    if ($kumpulanKegiatanPelatihan != null) {
        $skemBaru->addKumpulanAktivitas($kumpulanKegiatanPelatihan);
    }
```

```
        $kumpulanLkmmPemandu = (empty($request->lkmmPemandu))? null : $this->lkmmPemanduRepository->updateKumpulanLkmmPemandu($request->lkmmPemandu, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanLkmmPemandu != null) {
            $skemBaru->addKumpulanAktivitas($kumpulanLkmmPemandu);
        }

        $kumpulanPelatihan = (empty($request->pelatihan))? null : $this->pelatihanRepository->updateKumpulanPelatihan($request->pelatihan, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanPelatihan != null) {
            $skemBaru->addKumpulanAktivitas($kumpulanPelatihan);
        }

        $kumpulanAbdimas = (empty($request->abdimas))? null : $this->abdimasRepository->updateKumpulanAbdimas($request->abdimas, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimas != null) {
            $skemBaru->addKumpulanAktivitas($kumpulanAbdimas);
        }

        $kumpulanAbdimasWirausaha = (empty($request->abdimasWirausaha))? null : $this->abdimasWirausahaRepository->updateKumpulanAbdimas($request->abdimasWirausaha, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimasWirausaha != null) {
            $skemBaru->addKumpulanAktivitas($kumpulanAbdimasWirausaha);
        }

        $kumpulanAbdimasMagang = (empty($request->abdimasMagang))? null : $this->abdimasMagangRepository->updateKumpulanAbdimas($request->abdimasMagang, $idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanAbdimasMagang != null) {
```

```
            $skemBaru-
>addKumpulanAktivitas($kumpulanAbdimasMagang);
        }

        $kumpulanInternasionalisasi = (empty($request-
>internasionalisasi))? null : $this->internasionalisasiRepository-
>updateKumpulanInternasionalisasi($request->internasionalisasi,
$idMhs, $idSkem, date("Y-m-d"), $periode);
        if ($kumpulanInternasionalisasi != null) {
            $skemBaru-
>addKumpulanAktivitas($kumpulanInternasionalisasi);
        }

        $kumpulanPertukaranMhs = (empty($request->pertukaranMhs))?
null : $this->pertukaranMhsRepository-
>updateKumpulanPertukaranMhs($request->pertukaranMhs, $idMhs,
$idSkem, date("Y-m-d"), $periode);
        if ($kumpulanPertukaranMhs != null) {
            $skemBaru->addKumpulanAktivitas($kumpulanPertukaranMhs);
        }

        $kumpulanInternasionalisasiPelatihan = (empty($request-
>internasionalisasiPelatihan))? null : $this-
>internasionalisasiPelatihanRepository-
>updateKumpulanInternasionalisasi($request-
>internasionalisasiPelatihan, $idMhs, $idSkem, date("Y-m-d"),
$periode);
        if ($kumpulanInternasionalisasiPelatihan != null) {
            $skemBaru-
>addKumpulanAktivitas($kumpulanInternasionalisasiPelatihan);
        }


//        dd($skemBaru);

        $response = $this->skemRepository->update($skemBaru);
        $response = $skemBaru->getIdSkem()->id();
```

```
        return $response;
    }
}
```

5.2.2.17 Mahasiswa UpdateStatusSkemRequest

```
<?php

namespace
App\Modules\Skem\Application\Mahasiswa\UpdateStatusSkem;

class UpdateStatusSkemRequest
{
    public $idSkem;
    public $status;
    public $idUpdater;

    public function __construct($id_skem, $status, $id_updater)
    {
        $this->idSkem = $id_skem;
        $this->status = $status;
        $this->idUpdater = $id_updater;
    }
}
```

5.2.2.18 Mahasiswa UpdateStatuSkemService

```
<?php

namespace
App\Modules\Skem\Application\Mahasiswa\UpdateStatusSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\SkemRepository;

use
App\Modules\Skem\Application\Mahasiswa\UpdateStatusSkem\UpdateS
tatusSkemRequest;
```

```
class UpdateStatusSkemService
{
    private $skemRepository;

    public function __construct(SkemRepository $skemRepository)
    {
        $this->skemRepository = $skemRepository;
    }

    public function handle(UpdateStatusSkemRequest $request)
    {
        $idSkem = new SkemId($request->idSkem);
        $idUpdater = new MahasiswaId($request->idUpdater);

        $this->skemRepository->updateStatus($idSkem,     $request->status,
$idUpdater);
        return $request->idSkem;
    }
}
```

5.2.2.19 Validator UpdateStatusSkemRequest

```
<?php

namespace
App\Modules\Skem\Application\Validator\UpdateStatusSkem;

class UpdateStatusSkemRequest
{
    public $idSkem;
    public $status;
    public $idUpdater;
    public $message;

    public function __construct($idSkem, $status, $idUpdater, $message)
    {
        $this->idSkem = $idSkem;
```

```
        $this->status = $status;
        $this->idUpdater = $idUpdater;
        $this->message = $message;
    }
}
```

5.2.2.20 Validator UpdateStatusSKemService

```
<?php

namespace
App\Modules\Skem\Application\Validator\UpdateStatusSkem;

use App\Modules\Skem\Domain\Repositories\SkemRepository;

use
App\Modules\Skem\Application\Validator\UpdateStatusSkem\UpdateSta
tusAjuanSkemRequest;

class UpdateStatusSkemService
{
    private $skemRepository;

    public function __construct(SkemRepository $skemRepository)
    {
        $this->skemRepository = $skemRepository;
    }

    public function handle(UpdateStatusSkemRequest $request)
    {
        $this->skemRepository->updateStatusandMessage($request-
>idSkem, $request->status, $request->idUpdater, $request->message);
        return 0;
    }
}
```

5.2.2.21 Validator IndexPerkembanganSkemService

```
<?php
```

```
namespace
App\Modules\Skem\Application\Validator\IndexPerkembanganSkem;

use
App\Modules\Skem\Application\Validator\IndexPerkembanganSkem\Ind
exPerkembanganSkemResponse;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Validator\ValidatorId;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;
use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Repositories\SemesterRepository;

use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;

class IndexPerkembanganSkemService
{
   private $mahasiswaRepository;

   public function __construct(MahasiswaRepository
$mahasiswaRepository)
   {
      $this->mahasiswaRepository = $mahasiswaRepository;
   }
   public function handle($idSdm)
   {
      $idValidator = new ValidatorId($idSdm);

      $kumpulanAnakWali = $this->mahasiswaRepository-
>getKumpulanAnakWali($idValidator);

      $response = new
IndexPerkembanganSkemResponse($kumpulanAnakWali);
      return $response;
   }
}
```

5.2.2.22 Validator IndexperkembanganSkemResponse

```
<?php

namespace
App\Modules\Skem\Application\Validator\IndexPerkembanganSkem;

use App\Modules\Skem\Domain\Model\Mahasiswa;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use Mockery\Exception;

class IndexPerkembanganSkemResponse
{
   public $kumpulanAnakWali;
   public $jumlahAnakWali;

   public function __construct(array $kumpulanAnakWali)
   {
      $this->jumlahAnakWali = 0;
      $this->kumpulanAnakWali = [];

      foreach ($kumpulanAnakWali as $anakWali) {
         $nama = $anakWali->getNama();
         $thnAngkatan = $anakWali->getThnAngkatan();
         $nrp = $anakWali->getNrp();
         $idMhs = $anakWali->getIdMhs()->id();
         $pack = [
            'nama' => $nama,
            'thnAngkatan' => $thnAngkatan,
            'nrp' => $nrp,
            'idMhs' => $idMhs
         ];
         $this->kumpulanAnakWali[] = $pack;
         $this->jumlahAnakWali += 1;
      }
   }
}
```

5.2.3 Persistence

5.2.3.1 MssqlAbdimasMagangRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Abdimas\Abdimas;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\AbdimasMagangRepository;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;

use Illuminate\Support\Facades\DB;

class MssqlAbdimasMagangRepository implements
AbdimasMagangRepository
{
    const ID_JENIS_PORTOFOLIO = 'magang';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 8;
    const DURASI_MINIMUM_MAGANG = 5; //Dalam Bulan

    const ID_ASPEK_PELAKSANAAN = 21;
    const ID_ASPEK_RENTANG_WAKTU = 22;
    const ID_ASPEK_PERAN = 20;
```

```
    public function requestList( MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi', 'dbo.magang.id_jenis_dudi', '=',
'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang', 'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang', 'dbo.magang.id_skala_magang', '=',
'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
            ->where('dbo.magang.status_validasi', '=',
self::STATUS_VALIDASI)
            ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
            ->whereNotIn('dbo.magang.id_magang', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode)
            ->whereRaw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
dbo.magang.tgl_selesai) >= ' . self::DURASI_MINIMUM_MAGANG)

            ->select('dbo.magang.id_magang', 'dbo.magang.tempat',
'dbo.magang.deskripsi_kerja')
            ->get();
        return $kueriKumpulanMagang;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
```

```
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get( string $idMagang, MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi', 'dbo.magang.id_jenis_dudi', '=', 'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang', 'dbo.magang.id_kategori_magang', '=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang', 'dbo.magang.id_skala_magang', '=', 'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
            ->where('dbo.magang.id_magang', '=', $idMagang)
            ->where('dbo.magang.status_validasi', '=', self::STATUS_VALIDASI)
            ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
            ->whereNotIn('dbo.magang.id_magang', function($query) use ($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=', 'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=', self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=', self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai, GETDATE()) <= ' . $periode )
```

```
            ->whereRaw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
dbo.magang.tgl_selesai) >= ' . self::DURASI_MINIMUM_MAGANG)
            ->select('dbo.magang.*',
                'ref.jenis_dudi.nama as tempat_magang',
                'ref.kategori_magang.nama as kategori',
                'ref.skala_magang.nama as skala',
                DB::raw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
ISNULL(dbo.magang.tgl_selesai, GETDATE())) as rentang_waktu'))
            ->first();

        return $kueriMagang;
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu < 1 && $rentangWaktu > -1) {
            return 68;
        } elseif ($rentangWaktu <= 3) {
            return 69;
        } elseif ($rentangWaktu <= 6) {
            return 70;
        } elseif ($rentangWaktu > 6) {
            return 71;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'abdimas magang');
        }
    }

    public function formatPelaksanaan()
    {
        return 66; // otomatis individu
    }

//    public function formatPeran(int $idPeran)
//    {
//        if ($idPeran == 1) {
//            return 62; // anggota
//        } elseif ($idPeran == 2) {
```

```
//            return 63; // panitia inti
//        } elseif ($idPeran == 3) {
//            return 64; // asisten
//        } elseif ($idPeran == 4) {
//            return 65; // ketua
//        } else {
//            throw new PadananBobotNotFoundException('peran', 'abdimas');
//        }
//    }

    public function formatPeran (string $peran)
    {
        if ($peran == 'A') {
            return 62; // anggota
        } elseif ($peran == 'P') {
            return 63; // panitia inti
        } elseif ($peran == 'S') {
            return 64; // asisten
        } elseif ($peran == 'K') {
            return 65; // ketua
        } else {
            throw new PadananBobotNotFoundException('peran', 'abdimas magang');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int $idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai', '=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=', $idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
```

```
            'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
            'ref.kriteria_skem.*'
        )
        ->first();

    $kriteria = ElemenPenilaian::make(
        $kueriKriteria->id_kriteria_skem,
        $kueriKriteria->id_aspek_nilai,
        $kueriKriteria->aspek_nilai,
        $kueriKriteria->nama,
        $kueriKriteria->bobot
    );

    return $kriteria;
  }

  public function buildAbdimas(string $idMagang, MahasiswaId $idMhs,
SkemId $idSkem, string $tglKlaim, int $periode)
  {
    $kueriMagang = $this->get($idMagang, $idMhs, $periode);
    if ($kueriMagang == null) {
      throw new AktivitasIsNullException('magang');
    }
    $idKegiatanSkem = new AktivitasId();

    $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this-
>formatPeran('A'), self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
>formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriMagang->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $kumpulanElemenPenilaian = [$peran, $pelaksanaan,
$rentangWaktu];

    if($kueriMagang->jml_personel_penerima_manfaat == null) {
      $kueriMagang->jml_personel_penerima_manfaat = 1;
```

```
        }

        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriMagang->deskripsi_kerja,
            $kueriMagang->id_magang,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriMagang->jml_personel_penerima_manfaat
        );

        return $abdimas;
    }

    public function buildKumpulanAbdimas(array $kumpulanIdMagang,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdMagang as $idMagang) {
            $abdimas = $this->buildAbdimas($idMagang, $idMhs, $idSkem,
$tglKlaim, $periode);
            array_push($kumpulanAbdimas, $abdimas);
        }

        return $kumpulanAbdimas;
    }

    /*
     *  UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
```

```
        $idSkem = $idSkem->id();

        $kueriKumpulanMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi',          'dbo.magang.id_jenis_dudi',          '=',
'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang',    'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang',    'dbo.magang.id_skala_magang',    '=',
'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
            ->where('dbo.magang.status_validasi',                                '=',
self::STATUS_VALIDASI)
            ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.magang.id_magang',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode )
                        ->whereRaw('DATEDIFF(MONTH,
dbo.magang.tgl_mulai,        dbo.magang.tgl_selesai)        >=        '        .
self::DURASI_MINIMUM_MAGANG);
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.magang.id_magang',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
```

```
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.skem.id_skem', '=', $idSkem)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                });
//                  ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode );
            });
        })
        ->select('dbo.magang.id_magang',          'dbo.magang.tempat',
'dbo.magang.deskripsi_kerja')
        ->get();

    return $kueriKumpulanMagang;
  }

  public function getForUpdate(string $idMagang, MahasiswaId $idMhs,
int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriMagang = DB::table('dbo.magang')
        ->join('ref.jenis_dudi',        'dbo.magang.id_jenis_dudi',        '=',
'ref.jenis_dudi.id_jenis_dudi')
        ->join('ref.kategori_magang',   'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
        ->join('ref.skala_magang',    'dbo.magang.id_skala_magang',    '=',
'ref.skala_magang.id_skala_magang')
        ->where('dbo.magang.id_mhs', '=', $idMhs)
        ->where('dbo.magang.id_magang', '=', $idMagang)
        ->where('dbo.magang.status_validasi',                          '=',
self::STATUS_VALIDASI)
```

```
        ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
            $query0->where(function($query1) use ($idMhs, $periode) {
                $query1->whereNotIn('dbo.magang.id_magang',
function($query2) use ($idMhs) {
                    $query2->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
                })
                    ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode )
                    ->whereRaw('DATEDIFF(MONTH,
dbo.magang.tgl_mulai,        dbo.magang.tgl_selesai)        >=        '        .
self::DURASI_MINIMUM_MAGANG);
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.magang.id_magang',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                        });
//                      ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode );
                    });
        })
```

```
        ->select('dbo.magang.*',
            'ref.jenis_dudi.nama as tempat_magang',
            'ref.kategori_magang.nama as kategori',
            'ref.skala_magang.nama as skala',
            DB::raw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
ISNULL(dbo.magang.tgl_selesai, GETDATE())) as rentang_waktu'))
        ->first();

    return $kueriMagang;
  }

  public function updateAbdimas(string $idMagang, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
  {
    $kueriMagang = $this->getForUpdate($idMagang, $idMhs,
$periode, $idSkem);
    if ($kueriMagang == null) {
      throw new AktivitasIsNullException('magang');
    }
    $idKegiatanSkem = new AktivitasId();

    $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this-
>formatPeran('A'), self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
>formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriMagang->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $kumpulanElemenPenilaian = [$peran, $pelaksanaan,
$rentangWaktu];

    if($kueriMagang->jml_personel_penerima_manfaat == null) {
      $kueriMagang->jml_personel_penerima_manfaat = 1;
    }
```

```
        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriMagang->deskripsi_kerja,
            $kueriMagang->id_magang,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriMagang->jml_personel_penerima_manfaat
        );

        return $abdimas;
    }

    public function updateKumpulanAbdimas(array $kumpulanIdMagang,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdMagang as $idMagang) {
            $abdimas = $this->updateAbdimas($idMagang, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanAbdimas, $abdimas);
        }

        return $kumpulanAbdimas;
    }
}
```

5.2.3.2 MssqlAbdimasRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Abdimas\Abdimas;
```

```
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\AbdimasRepository;

use Illuminate\Support\Facades\DB;

class MssqlAbdimasRepository implements AbdimasRepository
{
   const ID_JENIS_PORTOFOLIO = 'abdimas';
   const STATUS_VALIDASI = 2;
   const ID_BIDANG_SKEM = 8;

   const ID_ASPEK_PELAKSANAAN = 21;
   const ID_ASPEK_RENTANG_WAKTU = 22;
   const ID_ASPEK_PERAN = 20;

   public function requestList(MahasiswaId $idMhs, int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriKumpulanAbdimas = DB::table('dbo.abdimas')
         ->join('dbo.anggota_abdimas', 'dbo.abdimas.id_abdimas', '=',
'dbo.anggota_abdimas.id_abdimas')
         ->where('dbo.anggota_abdimas.id_mhs', '=', $idMhs )
         ->where('dbo.abdimas.status_validasi', '=',
self::STATUS_VALIDASI)
         ->whereNotIn('dbo.abdimas.id_abdimas', function($query) use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
```

```
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.abdimas.nama', 'dbo.abdimas.id_abdimas')
            ->get();
        return $kueriKumpulanAbdimas;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idAbdimas, MahasiswaId $idMhs, int
$periode)
    {
        $idMhs = $idMhs->id();

        $kueriAbdimas = DB::table('dbo.abdimas')
            ->join('dbo.anggota_abdimas', 'dbo.abdimas.id_abdimas', '=',
'dbo.anggota_abdimas.id_abdimas')
            ->where('dbo.anggota_abdimas.id_mhs', '=', $idMhs )
            ->where('dbo.abdimas.id_abdimas', '=', $idAbdimas )
            ->where('dbo.abdimas.status_validasi', '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.abdimas.id_abdimas', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
```

```
        })
        ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode )
        ->select('dbo.abdimas.id_abdimas',
           'dbo.abdimas.nama',
           'dbo.abdimas.jml_masy_terdampak',
           'dbo.abdimas.is_regu',
           'dbo.abdimas.tgl_mulai',
           'dbo.abdimas.tgl_selesai',
           'dbo.abdimas.status_validasi',
           'dbo.abdimas.tgl_validasi',
           'dbo.abdimas.id_validator',
           'dbo.abdimas.nama_validator',
           'dbo.abdimas.catatan',
           'dbo.anggota_abdimas.id_mhs',
           'dbo.anggota_abdimas.peran',
           DB::raw('DATEDIFF(MONTH, dbo.abdimas.tgl_mulai,
            dbo.abdimas.tgl_selesai) as rentang_waktu') )
        ->first();

     return $kueriAbdimas;
  }

  public function formatPelaksanaan(int $isRegu)
  {
     if ($isRegu == 1)
     {
        return 67;
     } else {
        return 66;
     }
  }

  public function formatRentangWaktu(int $rentangWaktu)
  {
     if ($rentangWaktu < 1 && $rentangWaktu > -1) {
        return 68;
```

```
        } elseif ($rentangWaktu <= 3) {
            return 69;
        } elseif ($rentangWaktu <= 6) {
            return 70;
        } elseif ($rentangWaktu > 6) {
            return 71;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu', 'abdimas');
        }
    }

//    public function formatPeran(int $idPeran)
//    {
//        if ($idPeran == 1) {
//            return 62; // anggota
//        } elseif ($idPeran == 2) {
//            return 63; // panitia inti
//        } elseif ($idPeran == 3) {
//            return 64; // asisten
//        } elseif ($idPeran == 4) {
//            return 65; // ketua
//        } else {
//            throw new PadananBobotNotFoundException('peran', 'abdimas');
//        }
//    }

    public function formatPeran (string $peran)
    {
        if ($peran == 'A') {
            return 62; // anggota
        } elseif ($peran == 'P') {
            return 63; // panitia inti
        } elseif ($peran == 'S') {
            return 64; // asisten
        } elseif ($peran == 'K') {
            return 65; // ketua
        } else {
            throw new PadananBobotNotFoundException('peran', 'abdimas');
```

```
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildAbdimas($idAbdimas, MahasiswaId $idMhs,
SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriAbdimas = $this->get($idAbdimas, $idMhs, $periode);
        if ($kueriAbdimas == null) {
            throw new AktivitasIsNullException('abdimas');
        }
```

```
        $idKegiatanSkem = new AktivitasId();

        $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this->formatPeran($kueriAbdimas->peran), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $pelaksanaan = $this->getKriteria(self::ID_ASPEK_PELAKSANAAN, $this->formatPelaksanaan($kueriAbdimas->is_regu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriAbdimas->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$peran, $pelaksanaan, $rentangWaktu];


        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriAbdimas->nama,
            $kueriAbdimas->id_abdimas,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriAbdimas->jml_masy_terdampak
        );

        return $abdimas;
    }

    public function buildKumpulanAbdimas(array $kumpulanIdAbdimas, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdAbdimas as $idAbdimas) {
            $abdimas = $this->buildAbdimas($idAbdimas, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanAbdimas, $abdimas);
```

```
        }

        return $kumpulanAbdimas;
    }

    /*
     *  UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kumpulanKueriAbdimas = DB::table('dbo.abdimas')
            ->join('dbo.anggota_abdimas', 'dbo.abdimas.id_abdimas', '=',
'dbo.anggota_abdimas.id_abdimas')
            ->where('dbo.anggota_abdimas.id_mhs', '=', $idMhs )
            ->where('dbo.abdimas.status_validasi', '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.abdimas.id_abdimas',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                    })
                    ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode );
```

```
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.abdimas.id_abdimas',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                  ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.abdimas.nama', 'dbo.abdimas.id_abdimas')
        ->get();
    return $kumpulanKueriAbdimas;
  }

  public function getForUpdate(string $idAbdimas, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriAbdimas = DB::table('dbo.abdimas')
        ->join('dbo.anggota_abdimas',     'dbo.abdimas.id_abdimas',     '=',
'dbo.anggota_abdimas.id_abdimas')
        ->where('dbo.anggota_abdimas.id_mhs', '=', $idMhs )
        ->where('dbo.abdimas.id_abdimas', '=', $idAbdimas )
        ->where('dbo.abdimas.status_validasi',                              '=',
self::STATUS_VALIDASI)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
            $query0->where(function($query1) use ($idMhs, $periode) {
```

```
                $query1->whereNotIn('dbo.abdimas.id_abdimas',
function($query2) use ($idMhs) {
                    $query2->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                })
                    ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.abdimas.id_abdimas',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                        });
//                    ->whereRaw('DATEDIFF(DAY,dbo.abdimas.tgl_selesai,
GETDATE()) <= ' . $periode );
                    });
        })
        ->select('dbo.abdimas.id_abdimas',
            'dbo.abdimas.nama',
            'dbo.abdimas.jml_masy_terdampak',
            'dbo.abdimas.is_regu',
```

```
            'dbo.abdimas.tgl_mulai',
            'dbo.abdimas.tgl_selesai',
            'dbo.abdimas.status_validasi',
            'dbo.abdimas.tgl_validasi',
            'dbo.abdimas.id_validator',
            'dbo.abdimas.nama_validator',
            'dbo.abdimas.catatan',
            'dbo.anggota_abdimas.id_mhs',
            'dbo.anggota_abdimas.peran',
            DB::raw('DATEDIFF(MONTH, dbo.abdimas.tgl_mulai,
             dbo.abdimas.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriAbdimas;
  }

  public function updateAbdimas(string $idAbdimas, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
  {
    $kueriAbdimas = $this->getForUpdate($idAbdimas, $idMhs,
$periode, $idSkem);
    if ($kueriAbdimas == null) {
      throw new AktivitasIsNullException('abdimas');
    }
    $idKegiatanSkem = new AktivitasId();

    $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this-
>formatPeran($kueriAbdimas->peran), self::ID_BIDANG_SKEM,
$idKegiatanSkem);
    $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
>formatPelaksanaan($kueriAbdimas->is_regu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriAbdimas->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $kumpulanElemenPenilaian = [$peran, $pelaksanaan,
$rentangWaktu];
```

```
        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriAbdimas->nama,
            $kueriAbdimas->id_abdimas,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriAbdimas->jml_masy_terdampak
        );

        return $abdimas;
    }

    public function updateKumpulanAbdimas(array $kumpulanIdAbdimas,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdAbdimas as $idAbdimas) {
            $abdimas = $this->updateAbdimas($idAbdimas, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanAbdimas, $abdimas);
        }

        return $kumpulanAbdimas;
    }
}
```

5.2.3.3 MssqlAbdimasWirausahaRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
```

```
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Abdimas\Abdimas;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\AbdimasWirausahaRepository
;

use Illuminate\Support\Facades\DB;

class MssqlAbdimasWirausahaRepository implements
AbdimasWirausahaRepository
{
   const ID_JENIS_PORTOFOLIO = 'wirausaha';
   const STATUS_VALIDASI = 2;
   const ID_BIDANG_SKEM = 8;

   const ID_ASPEK_PELAKSANAAN = 21;
   const ID_ASPEK_RENTANG_WAKTU = 22;
   const ID_ASPEK_PERAN = 20;

   public function requestList(MahasiswaId $idMhs, int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriKumpulanWirausaha = DB::table('dbo.wirausaha')
         ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=',
'dbo.pelaku_usaha.id_wirausaha')
         ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
         ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs )
         ->where('dbo.pelaku_usaha.is_owner', '=', 1)
         ->where('dbo.wirausaha.status_validasi', '=',
self::STATUS_VALIDASI)
```

```
        ->where('dbo.lap_keuangan_usaha.status_validasi', '=',
self::STATUS_VALIDASI )
        ->whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
        })
        ->whereRaw('DATEPART(year,GETDATE()) -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365)
//          ->where(function ($query) use ($periode) {
//              $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                  ->orWhere(function($query) use ($periode) {
//                  $query->whereNotNull('dbo.wirausaha.tgl_tutup')
//                    ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//                  });
//          })
        ->select('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
'dbo.wirausaha.id_wirausaha',  'dbo.lap_keuangan_usaha.thn_laporan'  ,
'dbo.wirausaha.nama')
        ->get();

    return $kueriKumpulanWirausaha;
  }

  public function countRequestList(MahasiswaId $idMhs, int $periode)
  {
    $requestList = $this->requestList($idMhs, $periode);
    return count($requestList);
  }
```

```
   public function get(string $idLapkeuUsaha, MahasiswaId $idMhs, int
$periode)
   {
      $idMhs = $idMhs->id();

      $kueriWirausaha = DB::table('dbo.wirausaha')
         ->join('dbo.pelaku_usaha',     'dbo.wirausaha.id_wirausaha',     '=',
'dbo.pelaku_usaha.id_wirausaha')
         ->join('ref.jenis_badan_hukum',  'dbo.wirausaha.id_jenis_bh',  '=',
'ref.jenis_badan_hukum.id_jenis_bh')
         ->join('dbo.lap_keuangan_usaha',    'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
         ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs)
         ->where('dbo.pelaku_usaha.is_owner', '=', 1)
         ->where('dbo.lap_keuangan_usaha.id_lapkeu_usaha',              '=',
$idLapkeuUsaha)
         ->where('dbo.wirausaha.status_validasi',                       '=',
self::STATUS_VALIDASI)
         ->where('dbo.lap_keuangan_usaha.status_validasi',              '=',
self::STATUS_VALIDASI )
         ->whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem',       'dbo.kegiatan_skem.id_skem',       '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio',           '=',
self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem',                '=',
self::ID_BIDANG_SKEM);
         })
         ->whereRaw('DATEPART(year,GETDATE())                             -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365)
//          ->where(function ($query) use($periode) {
//             $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                ->orWhere(function($query) use($periode) {
//                   $query->whereNotNull('dbo.wirausaha.tgl_tutup')
```

```
//                  ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//              });
//          })
          ->latest('dbo.lap_keuangan_usaha.thn_laporan')
          ->select('dbo.wirausaha.id_wirausaha',
             'dbo.wirausaha.nama',
             'dbo.wirausaha.tgl_berdiri',
             'dbo.wirausaha.tempat_kedudukan',
             'dbo.wirausaha.bidang_usaha',
             'dbo.wirausaha.jml_modal_usaha',
             'dbo.wirausaha.tgl_tutup',
             'dbo.wirausaha.is_regu',
             'dbo.wirausaha.jml_pelaku',
             'dbo.wirausaha.jml_karyawan',
             'dbo.wirausaha.is_bidang_ilmu_berhubungan',
             'dbo.wirausaha.id_jenis_bh',
             'dbo.wirausaha.id_prog_wirausaha',
             'dbo.wirausaha.status_validasi',
             'dbo.wirausaha.tgl_validasi',
             'dbo.wirausaha.id_validator',
             'dbo.wirausaha.nama_validator',
             'dbo.wirausaha.catatan',
             'dbo.pelaku_usaha.id_mhs as id_mhs',
             'dbo.pelaku_usaha.is_owner as owner',
             'dbo.lap_keuangan_usaha.*',
             'ref.jenis_badan_hukum.nama as jenis_badan_hukum',
             DB::raw('DATEDIFF(MONTH,       dbo.wirausaha.tgl_berdiri,
ISNULL(dbo.wirausaha.tgl_tutup, GETDATE())) as rentang_waktu'))
          ->first();

      return $kueriWirausaha;
   }

   public function formatRentangWaktu(int $rentangWaktu)
   {
      if ($rentangWaktu < 1 && $rentangWaktu > -1) {
```

```
            return 68;
        } elseif ($rentangWaktu <= 3) {
            return 69;
        } elseif ($rentangWaktu <= 6) {
            return 70;
        } elseif ($rentangWaktu > 6) {
            return 71;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu', 'abdimas wirausaha');
        }
    }

    public function formatPelaksanaan(int $isRegu)
    {
        if ($isRegu == 1)
        {
            return 67;
        } else {
            return 66;
        }
    }

//    public function formatPeran(int $idPeran)
//    {
//        if ($idPeran == 1) {
//            return 62; // anggota
//        } elseif ($idPeran == 2) {
//            return 63; // panitia inti
//        } elseif ($idPeran == 3) {
//            return 64; // asisten
//        } elseif ($idPeran == 4) {
//            return 65; // ketua
//        } else {
//            throw new PadananBobotNotFoundException('peran', 'abdimas wirausaha');
//        }
//    }
```

```
    public function formatPeran (string $peran)
    {
        if ($peran == 'A') {
            return 62; // anggota
        } elseif ($peran == 'P') {
            return 63; // panitia inti
        } elseif ($peran == 'S') {
            return 64; // asisten
        } elseif ($peran == 'K') {
            return 65; // ketua
        } else {
            throw new PadananBobotNotFoundException('peran', 'abdimas
wirausaha');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
```

```
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildAbdimas(string $idLapkeuUsaha, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriWirausaha = $this->get($idLapkeuUsaha, $idMhs, $periode);
        if ($kueriWirausaha == null) {
            throw new AktivitasIsNullException('wirausaha');
        }
        $idKegiatanSkem = new AktivitasId();

        $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this-
>formatPeran('K'), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
>formatPelaksanaan($kueriWirausaha->is_regu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriWirausaha->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$peran, $pelaksanaan,
$rentangWaktu];


        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriWirausaha->nama,
            $kueriWirausaha->id_lapkeu_usaha,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriWirausaha->jml_karyawan
        );
```

```
        return $abdimas;
    }

    public function buildKumpulanAbdimas(array $kumpulanIdlapkeuUsaha, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdlapkeuUsaha as $idLapkeuUsaha) {
            $abdimas = $this->buildAbdimas($idLapkeuUsaha, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanAbdimas, $abdimas);
        }

        return $kumpulanAbdimas;
    }

    /*
     *  UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int $periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanWirausaha = DB::table('dbo.wirausaha')
            ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=', 'dbo.pelaku_usaha.id_wirausaha')
            ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha', '=', 'dbo.lap_keuangan_usaha.id_wirausaha')
            ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs )
            ->where('dbo.pelaku_usaha.is_owner', '=', 1)
            ->where('dbo.wirausaha.status_validasi', '=', self::STATUS_VALIDASI)
```

```
        ->where('dbo.lap_keuangan_usaha.status_validasi',          '=',
self::STATUS_VALIDASI )
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
          $query0->where(function($query1) use ($idMhs, $periode) {
            $query1-
>whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query2) use ($idMhs) {
              $query2->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem',   'dbo.kegiatan_skem.id_skem',   '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio',    '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem',        '=',
self::ID_BIDANG_SKEM);
            })
              ->whereRaw('DATEPART(year,GETDATE())                -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365);
          })
            ->orWhere(function($query3) use ($idMhs, $idSkem) {
              $query3-
>whereIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',  function($query4)
use ($idMhs, $idSkem) {
                $query4->select('dbo.kegiatan_skem.id_portofolio')
                  ->from('dbo.kegiatan_skem')
                  ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                  ->where('dbo.skem.id_mhs', '=', $idMhs)
                  ->where('dbo.skem.id_skem', '=', $idSkem)
                  ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                  ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
              });
//                                                                    -
>whereRaw('DATEDIFF(DAY,dbo.lap_keuangan_usaha.tgl_selesai,
GETDATE()) <= ' . $periode );
          });
```

```
        })
        ->select('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
'dbo.wirausaha.id_wirausaha', 'dbo.lap_keuangan_usaha.thn_laporan' ,
'dbo.wirausaha.nama')
        ->get();
      return $kueriKumpulanWirausaha;
   }

   public function getForUpdate(string $idLapkeuUsaha, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
   {
      $idMhs = $idMhs->id();
      $idSkem = $idSkem->id();

      $kueriWirausaha = DB::table('dbo.wirausaha')
        ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=',
'dbo.pelaku_usaha.id_wirausaha')
        ->join('ref.jenis_badan_hukum', 'dbo.wirausaha.id_jenis_bh', '=',
'ref.jenis_badan_hukum.id_jenis_bh')
        ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
        ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs)
        ->where('dbo.pelaku_usaha.is_owner', '=', 1)
        ->where('dbo.lap_keuangan_usaha.id_lapkeu_usaha', '=',
$idLapkeuUsaha)
        ->where('dbo.wirausaha.status_validasi', '=',
self::STATUS_VALIDASI)
        ->where('dbo.lap_keuangan_usaha.status_validasi', '=',
self::STATUS_VALIDASI )
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
          $query0->where(function($query1) use ($idMhs, $periode) {
            $query1-
>whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query2) use ($idMhs) {
              $query2->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
```

```
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                })
                  ->whereRaw('DATEPART(year,GETDATE()) -
dbo.lap_keuangan_usaha.thn_laporan <= ' . $periode/365);
            })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                $query3-
>whereIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha', function($query4)
use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                      ->from('dbo.kegiatan_skem')
                      ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                      ->where('dbo.skem.id_mhs', '=', $idMhs)
                      ->where('dbo.skem.id_skem', '=', $idSkem)
                      ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                      ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                  });
//                                                                  -
>whereRaw('DATEDIFF(DAY,dbo.lap_keuangan_usaha.tgl_selesai,
GETDATE()) <= ' . $periode );
              });
        })
//        ->where(function ($query) use($periode) {
//            $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                ->orWhere(function($query) use($periode) {
//                  $query->whereNotNull('dbo.wirausaha.tgl_tutup')
//                  ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//                });
//        })
```

```
            ->latest('dbo.lap_keuangan_usaha.thn_laporan')
            ->select('dbo.wirausaha.id_wirausaha',
                'dbo.wirausaha.nama',
                'dbo.wirausaha.tgl_berdiri',
                'dbo.wirausaha.tempat_kedudukan',
                'dbo.wirausaha.bidang_usaha',
                'dbo.wirausaha.jml_modal_usaha',
                'dbo.wirausaha.jml_karyawan',
                'dbo.wirausaha.tgl_tutup',
                'dbo.wirausaha.is_regu',
                'dbo.wirausaha.jml_pelaku',
                'dbo.wirausaha.is_bidang_ilmu_berhubungan',
                'dbo.wirausaha.id_jenis_bh',
                'dbo.wirausaha.id_prog_wirausaha',
                'dbo.wirausaha.status_validasi',
                'dbo.wirausaha.tgl_validasi',
                'dbo.wirausaha.id_validator',
                'dbo.wirausaha.nama_validator',
                'dbo.wirausaha.catatan',
                'dbo.pelaku_usaha.id_mhs as id_mhs',
                'dbo.pelaku_usaha.is_owner as owner',
                'dbo.lap_keuangan_usaha.*',
                'ref.jenis_badan_hukum.nama as jenis_badan_hukum',
                DB::raw('DATEDIFF(MONTH,       dbo.wirausaha.tgl_berdiri,
ISNULL(dbo.wirausaha.tgl_tutup, GETDATE())) as rentang_waktu'))
            ->first();

        return $kueriWirausaha;
    }

    public function updateAbdimas(string $idLapkeuUsaha, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriWirausaha = $this->getForUpdate($idLapkeuUsaha, $idMhs,
$periode, $idSkem);
        if ($kueriWirausaha == null) {
            throw new AktivitasIsNullException('wirausaha');
```

```
        }
        $idKegiatanSkem = new AktivitasId();

        $peran = $this->getKriteria(self::ID_ASPEK_PERAN, $this->formatPeran('K'), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $pelaksanaan = $this->getKriteria(self::ID_ASPEK_PELAKSANAAN, $this->formatPelaksanaan($kueriWirausaha->is_regu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriWirausaha->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$peran, $pelaksanaan, $rentangWaktu];


        $abdimas = Abdimas::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriWirausaha->nama,
            $kueriWirausaha->id_lapkeu_usaha,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriWirausaha->jml_karyawan
        );

        return $abdimas;
    }

    public function updateKumpulanAbdimas(array $kumpulanIdLapkeuUsaha, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanAbdimas = [];
        foreach ($kumpulanIdLapkeuUsaha as $idLapkeuUsaha) {
            $abdimas = $this->updateAbdimas($idLapkeuUsaha, $idMhs, $idSkem, $tglKlaim, $periode);
```

```
            array_push($kumpulanAbdimas, $abdimas);
        }

        return $kumpulanAbdimas;
    }
}
```

5.2.3.4 MssqlInternasionalisasiRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use
App\Modules\Skem\Domain\Model\Internasionalisasi\Internasionalisasi;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiRepository;

use Illuminate\Support\Facades\DB;

class MssqlInternasionalisasiRepository implements
InternasionalisasiRepository
{
    const DURASI_PORTOFOLIO = 365;
    const ID_JENIS_PORTOFOLIO = 'kegiatan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 11;

    const ID_ASPEK_PELAKSANAAN = 23;
```

```
    const ID_ASPEK_TINGKAT = 24;
    const ID_ASPEK_DISIPLIN = 25;
    const ID_ASPEK_RENTANG_WAKTU = 26;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 2, 3, 4])
            ->whereIn('dbo.kegiatan.id_skala_kegiatan', [4,5])
            ->where('dbo.kegiatan.status_validasi', '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.kegiatan.id_kegiatan', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
            ->get();
        return $kueriKumpulanKegiatan;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }
```

```
   public function get(string $idInternasionalisasi, MahasiswaId $idMhs,
int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriKegiatan = DB::table('dbo.kegiatan')
         ->join('ref.jenis_kegiatan',  'dbo.kegiatan.id_jenis_kegiatan',  '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
         ->join('ref.peran_kegiatan',  'dbo.kegiatan.id_peran_kegiatan',  '=',
'ref.peran_kegiatan.id_peran_kegiatan')
         ->join('ref.skala_kegiatan',  'dbo.kegiatan.id_skala_kegiatan',  '=',
'ref.skala_kegiatan.id_skala_kegiatan')
         ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 2, 3, 4])
         ->whereIn('dbo.kegiatan.id_skala_kegiatan', [4,5])
         ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
         ->where('dbo.kegiatan.id_kegiatan', '=', $idInternasionalisasi )
         ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
         ->whereNotIn('dbo.kegiatan.id_kegiatan',  function($query)  use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem',      'dbo.kegiatan_skem.id_skem',      '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio',          '=',
self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem',               '=',
self::ID_BIDANG_SKEM);
         })
         ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode )
         ->select('dbo.kegiatan.id_kegiatan',
            'dbo.kegiatan.nama',
            'dbo.kegiatan.tgl_mulai',
            'dbo.kegiatan.tgl_selesai',
            'dbo.kegiatan.deskripsi',
```

```
            'dbo.kegiatan.lokasi',
            'dbo.kegiatan.is_bidang_ilmu_berhubungan',
            'dbo.kegiatan.id_jenis_kegiatan',
            'dbo.kegiatan.id_peran_kegiatan',
            'dbo.kegiatan.id_skala_kegiatan',
            'dbo.kegiatan.status_validasi',
            'dbo.kegiatan.tgl_validasi',
            'dbo.kegiatan.id_validator',
            'dbo.kegiatan.nama_validator',
            'dbo.kegiatan.catatan',
            'dbo.kegiatan.id_mhs',
            'dbo.kegiatan.jml_anggota',
            'ref.jenis_kegiatan.nama as jenis_kegiatan',
            'ref.skala_kegiatan.nama as skala_kegiatan',
            'ref.peran_kegiatan.nama as posisi_kegiatan',
            DB::raw('DATEDIFF(MONTH, dbo.kegiatan.tgl_mulai,
             dbo.kegiatan.tgl_selesai) as rentang_waktu') )
        ->first();

      return $kueriKegiatan;
   }

//   public function formatPelaksanaan(int $idPelaksanaan)
//   {
//       switch ($idPelaksanaan) {
//           case 1: // individu
//               return 72;
//               break;
//           case 2: // berkelompok sd 3 orang
//               return 73;
//               break;
//           case 3: // berkelompok lebih dari 3 orang
//               return 74;
//               break;
//           default:
//                throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
//       }
//   }
```

```
    public function formatPelaksanaan(int $jumlahPelaksana)
    {
        if ($jumlahPelaksana < 1) {
            throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
        }
        if ($jumlahPelaksana == 1) {
            return 72; //individu
        } elseif ($jumlahPelaksana <= 3) {
            return 73; //berkelompok sd 3 orang
        } elseif ($jumlahPelaksana > 3) {
            return 74; //berkelompok lebih dari 3 orang
        } else {
            throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
        }

    }

    public function formatTingkat(int $idTingkat) {
        switch($idTingkat) {
            case 4: // nasional
                return 75;
                break;
            case 5: // internasional
                return 76;
                break;
            default:
                throw new PadananBobotNotFoundException('tingkat',
'internasionalisasi');
        }
    }

    public function formatBidangIlmu(int $bidangIlmu)
    {
        if ($bidangIlmu > 0) {
```

```
            return 78;
        } else {
            return 77;
        }
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu <= 1 && $rentangWaktu > -1) {
            return 79;
        } elseif ($rentangWaktu <= 3) {
            return 80;
        } elseif ($rentangWaktu <= 6) {
            return 81;
        } elseif ($rentangWaktu <= 12) {
            return 82;
        } elseif ($rentangWaktu > 12) {
            return 83;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'internasionalisasi');
        }
    }

//    public function formatPosisi(string $posisi)
//    {
//        if  ($posisi == 'Anggota Panitia' || $posisi == 'Panitia Inti') {
//            return "Panitia";
//        } else {
//            return $posisi;
//        }
//    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
```

```
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildInternasionalisasi(string $idInternasionalisasi,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->get($idInternasionalisasi, $idMhs,
$periode);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('kegiatan');
        }
        $idKegiatanSkem = new AktivitasId();

        $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
>formatPelaksanaan($kueriKegiatan->jml_anggota),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this->formatTingkat($kueriKegiatan->id_skala_kegiatan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this->formatBidangIlmu($kueriKegiatan->is_bidang_ilmu_berhubungan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriKegiatan->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin, $rentangWaktu];

        $kegiatan = Internasionalisasi::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim, $kueriKegiatan->nama,
            $kueriKegiatan->id_kegiatan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriKegiatan->id_peran_kegiatan
        );

        return $kegiatan;
    }

    public function buildKumpulanInternasionalisasi(array $kumpulanIdInternasionalisasi, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdInternasionalisasi as $idInternasionalisasi) {
            $kegiatan = $this->buildInternasionalisasi($idInternasionalisasi, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanKegiatan, $kegiatan);
        }

        return $kumpulanKegiatan;
    }
```

```
   /*
    * UPDATE
    */

   public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
   {
      $idMhs = $idMhs->id();
      $idSkem = $idSkem->id();

      $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
         ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
         ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 2, 3, 4])
         ->whereIn('dbo.kegiatan.id_skala_kegiatan', [4,5])
         ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
         ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
            $query0->where(function($query1) use ($idMhs, $periode) {
               $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                  $query2->select('dbo.kegiatan_skem.id_portofolio')
                     ->from('dbo.kegiatan_skem')
                     ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                     ->where('dbo.skem.id_mhs', '=', $idMhs)
                     ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                     ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
               })
                  ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
               ->orWhere(function($query3) use ($idMhs, $idSkem) {
```

```
                $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                });
//                  ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
            });
        })
        ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
        ->get();
    return $kueriKumpulanKegiatan;
  }

  public function getForUpdate(string $idInternasionalisasi,
MahasiswaId $idMhs, int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriKegiatan = DB::table('dbo.kegiatan')
        ->join('ref.jenis_kegiatan',   'dbo.kegiatan.id_jenis_kegiatan',   '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
        ->join('ref.peran_kegiatan',  'dbo.kegiatan.id_peran_kegiatan',  '=',
'ref.peran_kegiatan.id_peran_kegiatan')
        ->join('ref.skala_kegiatan',   'dbo.kegiatan.id_skala_kegiatan',   '=',
'ref.skala_kegiatan.id_skala_kegiatan')
        ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 2, 3, 4])
        ->whereIn('dbo.kegiatan.id_skala_kegiatan', [4,5])
        ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
        ->where('dbo.kegiatan.id_kegiatan', '=', $idInternasionalisasi )
```

```
        ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
           $query0->where(function($query1) use ($idMhs, $periode) {
              $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                 $query2->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
              })
                 ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
           })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                 $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                       ->from('dbo.kegiatan_skem')
                       ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                       ->where('dbo.skem.id_mhs', '=', $idMhs)
                       ->where('dbo.skem.id_skem', '=', $idSkem)
                       ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                       ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                 });
//                  ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
              });
        })
```

```
            ->select('dbo.kegiatan.id_kegiatan',
                'dbo.kegiatan.nama',
                'dbo.kegiatan.tgl_mulai',
                'dbo.kegiatan.tgl_selesai',
                'dbo.kegiatan.deskripsi',
                'dbo.kegiatan.lokasi',
                'dbo.kegiatan.is_bidang_ilmu_berhubungan',
                'dbo.kegiatan.id_jenis_kegiatan',
                'dbo.kegiatan.id_peran_kegiatan',
                'dbo.kegiatan.id_skala_kegiatan',
                'dbo.kegiatan.status_validasi',
                'dbo.kegiatan.tgl_validasi',
                'dbo.kegiatan.id_validator',
                'dbo.kegiatan.nama_validator',
                'dbo.kegiatan.catatan',
                'dbo.kegiatan.id_mhs',
                'dbo.kegiatan.jml_anggota',
                'ref.jenis_kegiatan.nama as jenis_kegiatan',
                'ref.skala_kegiatan.nama as skala_kegiatan',
                'ref.peran_kegiatan.nama as posisi_kegiatan',
                DB::raw('DATEDIFF(MONTH, dbo.kegiatan.tgl_mulai,
                 dbo.kegiatan.tgl_selesai) as rentang_waktu') )
            ->first();

        return $kueriKegiatan;
    }

    public function updateInternasionalisasi(string $idInternasionalisasi,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->getForUpdate($idInternasionalisasi,
$idMhs, $periode, $idSkem);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('kegiatan');
        }
        $idKegiatanSkem = new AktivitasId();

        $pelaksanaan = $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN, $this-
```

```
>formatPelaksanaan($kueriKegiatan->jml_anggota),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this-
>formatTingkat($kueriKegiatan->id_skala_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this-
>formatBidangIlmu($kueriKegiatan->is_bidang_ilmu_berhubungan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriKegiatan->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin,
$rentangWaktu];

      $kegiatan = Internasionalisasi::make(
         $idKegiatanSkem,
         self::ID_BIDANG_SKEM,
         $tglKlaim, $kueriKegiatan->nama,
         $kueriKegiatan->id_kegiatan,
         self::ID_JENIS_PORTOFOLIO,
         $kumpulanElemenPenilaian,
         $kueriKegiatan->id_peran_kegiatan
      );

      return $kegiatan;
   }

   public function updateKumpulanInternasionalisasi(array
$kumpulanIdInternasionalisasi, MahasiswaId $idMhs, SkemId $idSkem,
string $tglKlaim, int $periode)
   {
      $kegiatans = [];
      foreach ($kumpulanIdInternasionalisasi as $idInternasionalisasi) {
         $kegiatan = $this->updateInternasionalisasi($idInternasionalisasi,
$idMhs, $idSkem, $tglKlaim, $periode);
         array_push($kegiatans, $kegiatan);
```

```
        }

        return $kegiatans;
    }


}
```

5.2.3.5 MssqlInternasionalisasiPelatihanRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use
App\Modules\Skem\Domain\Model\Internasionalisasi\Internasionalisasi;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\InternasionalisasiPelatihanRep
ository;

use Illuminate\Support\Facades\DB;

class MssqlInternasionalisasiPelatihanRepository implements
InternasionalisasiPelatihanRepository
{
    const DURASI_PORTOFOLIO = 365;
    const ID_JENIS_PORTOFOLIO = 'pelatihan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 11;

    const ID_ASPEK_PELAKSANAAN = 23;
```

```
    const ID_ASPEK_TINGKAT = 24;
    const ID_ASPEK_DISIPLIN = 25;
    const ID_ASPEK_RENTANG_WAKTU = 26;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih',
'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan', 'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi', '=',
self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [4,5])
            ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode)
            ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
            ->get();
        return $kueriKumpulanPelatihan;
    }
```

```
    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idPelatihan, MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih', '=', 'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan', 'dbo.pelatihan.id_skala_pelatihan', 'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
            ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
            ->where('dbo.pelatihan.status_validasi', '=', self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [4,5])
            ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use ($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=', 'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=', self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=', self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai, GETDATE()) <= ' . $periode )
            ->select('dbo.pelatihan.id_pelatihan',
                'dbo.pelatihan.nama',
                'dbo.pelatihan.tgl_mulai',
```

```
            'dbo.pelatihan.tgl_selesai',
            'dbo.pelatihan.penyelenggara',
            'dbo.pelatihan.id_jenis_latih',
            'dbo.pelatihan.id_skala_pelatihan',
            'dbo.pelatihan.status_validasi',
            'dbo.pelatihan.tgl_validasi',
            'dbo.pelatihan.id_validator',
            'dbo.pelatihan.nama_validator',
            'dbo.pelatihan.catatan',
            'dbo.pelatihan.id_mhs',
            'dbo.pelatihan.is_bidang_ilmu_berhubungan',
            'ref.jenis_pelatihan.nama as jenis_pelatihan',
            'ref.skala_pelatihan.nama as skala_pelatihan',
            DB::raw('DATEDIFF(MONTH, dbo.pelatihan.tgl_mulai,
             dbo.pelatihan.tgl_selesai) as rentang_waktu') )
        ->first();

      return $kueriPelatihan;
   }

//   public function formatPelaksanaan(int $idPelaksanaan)
//   {
//       switch ($idPelaksanaan) {
//           case 1: // individu
//               return 72;
//               break;
//           case 2: // berkelompok sd 3 orang
//               return 73;
//               break;
//           case 3: // berkelompok lebih dari 3 orang
//               return 74;
//               break;
//           default:
//                throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
//       }
//   }
```

```
//    public function formatPelaksanaan(int $jumlahPelaksana)
//    {
//        if ($jumlahPelaksana < 1) {
//                throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
//        }
//        if ($jumlahPelaksana == 1) {
//            return 72; //individu
//        } elseif ($jumlahPelaksana <= 3) {
//            return 73; //berkelompok sd 3 orang
//        } elseif ($jumlahPelaksana > 3) {
//            return 74; //berkelompok lebih dari 3 orang
//        } else {
//                throw new PadananBobotNotFoundException('pelaksanaan',
'internasionalisasi');
//        }
//    }
    public function formatPelaksanaan()
    {
        return 72; //otomatis individu
    }

    public function formatTingkat(int $idTingkat) {
        switch($idTingkat) {
            case 4: // nasional
                return 75;
                break;
            case 5: // internasional
                return 76;
                break;
            default:
                throw       new       PadananBobotNotFoundException('tingkat',
'internasionalisasi');
        }
    }

    public function formatBidangIlmu(int $bidangIlmu)
    {
```

```
        if ($bidangIlmu > 0) {
            return 78;
        } else {
            return 77;
        }
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu <= 1 && $rentangWaktu > -1) {
            return 79;
        } elseif ($rentangWaktu <= 3) {
            return 80;
        } elseif ($rentangWaktu <= 6) {
            return 81;
        } elseif ($rentangWaktu <= 12) {
            return 82;
        } elseif ($rentangWaktu > 12) {
            return 83;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'internasionalisasi');
        }
    }

//    public function formatPosisi(string $posisi)
//    {
//        if  ($posisi == 'Anggota Panitia' || $posisi == 'Panitia Inti') {
//            return "Panitia";
//        } else {
//            return $posisi;
//        }
//    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
```

```
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem',    'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem',                '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildInternasionalisasi(string $idPelatihan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->get($idPelatihan, $idMhs, $periode);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('pelatihan
internasionalisasi');
        }
        $idKegiatanSkem = new AktivitasId();

        $pelaksanaan                =                $this-
>getKriteria(self::ID_ASPEK_PELAKSANAAN,                $this-
>formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this->formatTingkat($kueriKegiatan->id_skala_pelatihan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this->formatBidangIlmu($kueriKegiatan->is_bidang_ilmu_berhubungan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriKegiatan->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin, $rentangWaktu];

        $kegiatan = Internasionalisasi::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim, $kueriKegiatan->nama,
            $kueriKegiatan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            1
        );

        return $kegiatan;
    }

    public function buildKumpulanInternasionalisasi(array $kumpulanIdPelatihan, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $kegiatan = $this->buildInternasionalisasi($idPelatihan, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanKegiatan, $kegiatan);
        }
```

```
        return $kumpulanKegiatan;
    }


    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan',     'dbo.pelatihan.id_jenis_latih',     '=',
'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan',       'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi',                              '=',
self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [4, 5])
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
```

```
                ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                  $query4->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.skem.id_skem', '=', $idSkem)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                });
//                ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
              });
        })
        ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
        ->get();
      return $kueriKumpulanPelatihan;
  }

  public function getForUpdate(string $idPelatihan, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriPelatihan = DB::table('dbo.pelatihan')
      ->join('ref.jenis_pelatihan',     'dbo.pelatihan.id_jenis_latih',     '=',
'ref.jenis_pelatihan.id_jenis_latih')
      ->join('ref.skala_pelatihan',     'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
```

```
        ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
        ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
        ->where('dbo.pelatihan.status_validasi',                    '=',
self::STATUS_VALIDASI)
        ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
        ->whereIn('dbo.pelatihan.id_skala_pelatihan', [4,5])
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
           $query0->where(function($query1) use ($idMhs, $periode) {
              $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
                 $query2->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
              })
                 ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
           })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                 $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                       ->from('dbo.kegiatan_skem')
                       ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                       ->where('dbo.skem.id_mhs', '=', $idMhs)
                       ->where('dbo.skem.id_skem', '=', $idSkem)
                       ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                       ->where('dbo.kegiatan_skem.id_bidang_skem',      '=',
self::ID_BIDANG_SKEM);
                 });
//                 ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
```

```
                });
            })
            ->select('dbo.pelatihan.id_pelatihan',
                'dbo.pelatihan.nama',
                'dbo.pelatihan.tgl_mulai',
                'dbo.pelatihan.tgl_selesai',
                'dbo.pelatihan.penyelenggara',
                'dbo.pelatihan.id_jenis_latih',
                'dbo.pelatihan.id_skala_pelatihan',
                'dbo.pelatihan.status_validasi',
                'dbo.pelatihan.tgl_validasi',
                'dbo.pelatihan.id_validator',
                'dbo.pelatihan.nama_validator',
                'dbo.pelatihan.catatan',
                'dbo.pelatihan.id_mhs',
                'dbo.pelatihan.is_bidang_ilmu_berhubungan',
                'ref.jenis_pelatihan.nama as jenis_pelatihan',
                'ref.skala_pelatihan.nama as skala_pelatihan',
                DB::raw('DATEDIFF(MONTH, dbo.pelatihan.tgl_mulai,
                 dbo.pelatihan.tgl_selesai) as rentang_waktu') )
            ->first();

        return $kueriPelatihan;
    }

    public function updateInternasionalisasi(string $idPelatihan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->getForUpdate($idPelatihan, $idMhs,
$periode, $idSkem);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('pelatihan
internasionalisasi');
        }
        $idKegiatanSkem = new AktivitasId();
```

```
        $pelaksanaan = $this->getKriteria(self::ID_ASPEK_PELAKSANAAN, $this->formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this->formatTingkat($kueriKegiatan->id_skala_pelatihan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this->formatBidangIlmu($kueriKegiatan->is_bidang_ilmu_berhubungan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriKegiatan->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin, $rentangWaktu];

        $kegiatan = Internasionalisasi::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim, $kueriKegiatan->nama,
            $kueriKegiatan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            1
        );

        return $kegiatan;
    }

    public function updateKumpulanInternasionalisasi(array $kumpulanIdPelatihan, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kegiatans = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $kegiatan = $this->updateInternasionalisasi($idPelatihan, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kegiatans, $kegiatan);
        }
```

```
        return $kegiatans;
    }


}
```

5.2.3.6 MssqlKegiatanRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Kegiatan\Kegiatan;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\KegiatanRepository;

use Illuminate\Support\Facades\DB;

class MssqlKegiatanRepository implements KegiatanRepository
{
    const ID_JENIS_PORTOFOLIO = 'kegiatan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 7;

    const ID_ASPEK_SKALA = 17;
    const ID_ASPEK_POSISI = 18;
    const ID_ASPEK_RENTANG_WAKTU = 19;

    public function requestList(MahasiswaId $idMhs, int $periode)
```

```
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 3, 4, 9999])
            ->whereIn('dbo.kegiatan.id_skala_kegiatan', [1,2,3])
            ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.kegiatan.id_kegiatan',   function($query)   use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',        'dbo.kegiatan_skem.id_skem',        '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',            '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',                 '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode)
            ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
            ->get();
        return $kueriKumpulanKegiatan;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idKegiatan, MahasiswaId $idMhs, int
$periode)
    {
        $idMhs = $idMhs->id();
```

```
        $kueriKegiatan = DB::table('dbo.kegiatan')
            ->join('ref.jenis_kegiatan',  'dbo.kegiatan.id_jenis_kegiatan',  '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
            ->join('ref.peran_kegiatan',  'dbo.kegiatan.id_peran_kegiatan',  '=',
'ref.peran_kegiatan.id_peran_kegiatan')
            ->join('ref.skala_kegiatan',  'dbo.kegiatan.id_skala_kegiatan',  '=',
'ref.skala_kegiatan.id_skala_kegiatan')
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 3, 4, 9999])
            ->whereIn('dbo.kegiatan.id_skala_kegiatan', [1,2,3])
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
            ->where('dbo.kegiatan.id_kegiatan', '=', $idKegiatan )
            ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.kegiatan.id_kegiatan',  function($query)  use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',       'dbo.kegiatan_skem.id_skem',       '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',           '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',                '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.kegiatan.id_kegiatan',
                'dbo.kegiatan.nama',
                'dbo.kegiatan.tgl_mulai',
                'dbo.kegiatan.tgl_selesai',
                'dbo.kegiatan.deskripsi',
                'dbo.kegiatan.lokasi',
                'dbo.kegiatan.is_bidang_ilmu_berhubungan',
                'dbo.kegiatan.id_jenis_kegiatan',
                'dbo.kegiatan.id_peran_kegiatan',
                'dbo.kegiatan.id_skala_kegiatan',
```

```
            'dbo.kegiatan.status_validasi',
            'dbo.kegiatan.tgl_validasi',
            'dbo.kegiatan.id_validator',
            'dbo.kegiatan.nama_validator',
            'dbo.kegiatan.catatan',
            'dbo.kegiatan.id_mhs',
            'ref.jenis_kegiatan.nama as jenis_kegiatan',
            'ref.skala_kegiatan.nama as skala_kegiatan',
            'ref.peran_kegiatan.nama as posisi_kegiatan',
            DB::raw('DATEDIFF(DAY, dbo.kegiatan.tgl_mulai,
             dbo.kegiatan.tgl_selesai) as rentang_waktu') )
        ->first();
    return $kueriKegiatan;
}

public function formatSkala($idSkala)
{
    if ($idSkala == 1) {
        return 53; // departemen
    } elseif ($idSkala == 2) {
        return 54; // fakultas
    } elseif ($idSkala == 3) {
        return 55; // institut
    } else {
        throw new PadananBobotNotFoundException('skala', 'kegiatan');
    }
}

public function formatPosisi($idPosisi)
{
    if ($idPosisi == 1) {
        return 56; // peserta
    } elseif ($idPosisi == 2) {
        return 57; // anggota panitia
    } elseif ($idPosisi == 3 || $idPosisi == 4) {
        return 58; // panitia inti || pembicara
    } else {
        throw new PadananBobotNotFoundException('posisi', 'kegiatan');
    }
```

```
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu <= 1 && $rentangWaktu > -1) {
            return 59; // <=1 hari
        } elseif ($rentangWaktu <= 3) {
            return 60; // 2-3 hari
        } elseif ($rentangWaktu > 3) {
            return 61; // > 3 hari
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'kegiatan');
        }
    }


    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
```

```
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildKegiatan(string $idKegiatan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->get($idKegiatan, $idMhs, $periode);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('kegiatan');
        }
        $idKegiatanSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriKegiatan->id_skala_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $posisi = $this->getKriteria(self::ID_ASPEK_POSISI, $this-
>formatPosisi($kueriKegiatan->id_peran_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriKegiatan->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$skala, $posisi, $rentangWaktu];

        $kegiatan = Kegiatan::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriKegiatan->nama,
            $kueriKegiatan->id_kegiatan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );
        return $kegiatan;
```

```
    }

    public function buildKumpulanKegiatan(array $kumpulanIdKegiatan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdKegiatan as $idKegiatan) {
            $kegiatan = $this->buildKegiatan($idKegiatan, $idMhs, $idSkem,
$tglKlaim, $periode);
            array_push($kumpulanKegiatan, $kegiatan);
        }
        return $kumpulanKegiatan;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 3, 4, 9999])
            ->whereIn('dbo.kegiatan.id_skala_kegiatan', [1,2,3])
            ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
```

```
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                })
                ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
            ->orWhere(function($query3) use ($idMhs, $idSkem) {
                $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.skem.id_skem', '=', $idSkem)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                });
//                ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
            });
        })
        ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
        ->get();
    return $kueriKumpulanKegiatan;
  }

  public function getForUpdate(string $idKegiatan, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();
```

```
        $kueriKegiatan = DB::table('dbo.kegiatan')
            ->join('ref.jenis_kegiatan',   'dbo.kegiatan.id_jenis_kegiatan',   '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
            ->join('ref.peran_kegiatan',  'dbo.kegiatan.id_peran_kegiatan',  '=',
'ref.peran_kegiatan.id_peran_kegiatan')
            ->join('ref.skala_kegiatan',   'dbo.kegiatan.id_skala_kegiatan',   '=',
'ref.skala_kegiatan.id_skala_kegiatan')
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [1, 3, 4, 9999])
            ->whereIn('dbo.kegiatan.id_skala_kegiatan', [1,2,3])
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
            ->where('dbo.kegiatan.id_kegiatan', '=', $idKegiatan )
            ->where('dbo.kegiatan.status_validasi',                              '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
               $query0->where(function($query1) use ($idMhs, $periode) {
                  $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                     $query2->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                  })
                     ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
               })
                  ->orWhere(function($query3) use ($idMhs, $idSkem) {
                     $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                           ->from('dbo.kegiatan_skem')
```

```
                        ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                   ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.kegiatan.id_kegiatan',
            'dbo.kegiatan.nama',
            'dbo.kegiatan.tgl_mulai',
            'dbo.kegiatan.tgl_selesai',
            'dbo.kegiatan.deskripsi',
            'dbo.kegiatan.lokasi',
            'dbo.kegiatan.is_bidang_ilmu_berhubungan',
            'dbo.kegiatan.id_jenis_kegiatan',
            'dbo.kegiatan.id_peran_kegiatan',
            'dbo.kegiatan.id_skala_kegiatan',
            'dbo.kegiatan.status_validasi',
            'dbo.kegiatan.tgl_validasi',
            'dbo.kegiatan.id_validator',
            'dbo.kegiatan.nama_validator',
            'dbo.kegiatan.catatan',
            'dbo.kegiatan.id_mhs',
            'ref.jenis_kegiatan.nama as jenis_kegiatan',
            'ref.skala_kegiatan.nama as skala_kegiatan',
            'ref.peran_kegiatan.nama as posisi_kegiatan',
            DB::raw('DATEDIFF(DAY, dbo.kegiatan.tgl_mulai,
             dbo.kegiatan.tgl_selesai) as rentang_waktu') )
        ->first();
    return $kueriKegiatan;
  }
```

```
    public function updateKegiatan(string $idKegiatan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriKegiatan = $this->getForUpdate($idKegiatan, $idMhs,
$periode, $idSkem);
        if ($kueriKegiatan == null) {
            throw new AktivitasIsNullException('kegiatan');
        }
        $idKegiatanSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriKegiatan->id_skala_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $posisi = $this->getKriteria(self::ID_ASPEK_POSISI, $this-
>formatPosisi($kueriKegiatan->id_peran_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriKegiatan->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$skala, $posisi, $rentangWaktu];

        $kegiatan = Kegiatan::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriKegiatan->nama,
            $kueriKegiatan->id_kegiatan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );
        return $kegiatan;
    }

    public function updateKumpulanKegiatan(array $kumpulanIdKegiatan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
```

```
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdKegiatan as $idKegiatan) {
            $kegiatan = $this->updateKegiatan($idKegiatan, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanKegiatan, $kegiatan);
        }
        return $kumpulanKegiatan;
    }

}
```

5.2.3.7 MssqlKegiatanPelatihanRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Kegiatan\Kegiatan;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\KegiatanPelatihanRepository;
use Illuminate\Support\Facades\DB;

class MssqlKegiatanPelatihanRepository implements
KegiatanPelatihanRepository
{
    const ID_JENIS_PORTOFOLIO = 'pelatihan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 7;

    const ID_ASPEK_SKALA = 17;
```

```
    const ID_ASPEK_POSISI = 18;
    const ID_ASPEK_RENTANG_WAKTU = 19;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih',
'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan', 'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi', '=',
self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [1,2,3])
            ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode)
            ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
            ->get();
        return $kueriKumpulanPelatihan;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
```

```
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idPelatihan, MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih', '=', 'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan', 'dbo.pelatihan.id_skala_pelatihan', 'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
            ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
            ->where('dbo.pelatihan.status_validasi', '=', self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [1,2,3])
            ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use ($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=', 'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=', self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=', self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai, GETDATE()) <= ' . $periode )
            ->select('dbo.pelatihan.id_pelatihan',
                'dbo.pelatihan.nama',
                'dbo.pelatihan.tgl_mulai',
                'dbo.pelatihan.tgl_selesai',
```

```
            'dbo.pelatihan.penyelenggara',
            'dbo.pelatihan.id_jenis_latih',
            'dbo.pelatihan.id_skala_pelatihan',
            'dbo.pelatihan.status_validasi',
            'dbo.pelatihan.tgl_validasi',
            'dbo.pelatihan.id_validator',
            'dbo.pelatihan.nama_validator',
            'dbo.pelatihan.catatan',
            'dbo.pelatihan.id_mhs',
            'ref.jenis_pelatihan.nama as jenis_pelatihan',
            'ref.skala_pelatihan.nama as skala_pelatihan',
            DB::raw('DATEDIFF(DAY, dbo.pelatihan.tgl_mulai,
             dbo.pelatihan.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriPelatihan;
  }

  public function formatSkala($idSkala)
  {
    if ($idSkala == 1) {
      return 53; // departemen
    } elseif ($idSkala == 2) {
      return 54; // fakultas
    } elseif ($idSkala == 3) {
      return 55; // institut
    } elseif ($idSkala == 4) {
      return 55; // nasional
    } elseif ($idSkala == 5) {
      return 55; // internasional
    }  else {
      throw new PadananBobotNotFoundException('skala', 'kegiatan
pelatihan');
    }
  }

  public function formatRentangWaktu(int $rentangWaktu)
```

```
    {
        if ($rentangWaktu <= 1 && $rentangWaktu > -1) {
            return 59; // <=1 hari
        } elseif ($rentangWaktu <= 3) {
            return 60; // 2-3 hari
        } elseif ($rentangWaktu > 3) {
            return 61; // > 3 hari
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'kegiatan pelatihan');
        }
    }


    public function formatPosisi()
    {
        return 56; // otomatis peserta
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
```

```
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildKegiatan(string $idPelatihan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriPelatihan = $this->get($idPelatihan, $idMhs, $periode);
        if ($kueriPelatihan == null) {
            throw new AktivitasIsNullException('kegiatan pelatihan');
        }
        $idkegiatanSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriPelatihan->id_skala_pelatihan),
self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $posisi = $this->getKriteria(self::ID_ASPEK_POSISI, $this-
>formatPosisi(), self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriPelatihan->rentang_waktu),
self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $kumpulanElemenPenilaian = [$skala, $posisi, $rentangWaktu];

        $kegiatan = Kegiatan::make(
            $idkegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim, $kueriPelatihan->nama,
            $kueriPelatihan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );
        return $kegiatan;
```

```
    }

    public function buildKumpulanKegiatan(array $kumpulanIdPelatihan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $kegiatan = $this->buildKegiatan($idPelatihan, $idMhs, $idSkem,
$tglKlaim, $periode);
            array_push($kumpulanKegiatan, $kegiatan);
        }
        return $kumpulanKegiatan;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan',     'dbo.pelatihan.id_jenis_latih',     '=',
'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan',       'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi',                              '=',
self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [1,2,3])
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
```

```
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
                })
                  ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                  $query4->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.skem.id_skem', '=', $idSkem)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                  });
//                ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
              });
        })
        ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
        ->get();
    return $kueriKumpulanPelatihan;
  }

  public function getForUpdate(string $idPelatihan, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
```

```
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan',     'dbo.pelatihan.id_jenis_latih',     '=',
'ref.jenis_pelatihan.id_jenis_latih')
            ->join('ref.skala_pelatihan',       'dbo.pelatihan.id_skala_pelatihan',
'ref.skala_pelatihan.id_skala_pelatihan')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
            ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
            ->where('dbo.pelatihan.status_validasi',                                '=',
self::STATUS_VALIDASI)
            ->whereIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereIn('dbo.pelatihan.id_skala_pelatihan', [1,2,3])
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
```

```
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                  ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.pelatihan.id_pelatihan',
            'dbo.pelatihan.nama',
            'dbo.pelatihan.tgl_mulai',
            'dbo.pelatihan.tgl_selesai',
            'dbo.pelatihan.penyelenggara',
            'dbo.pelatihan.id_jenis_latih',
            'dbo.pelatihan.id_skala_pelatihan',
            'dbo.pelatihan.status_validasi',
            'dbo.pelatihan.tgl_validasi',
            'dbo.pelatihan.id_validator',
            'dbo.pelatihan.nama_validator',
            'dbo.pelatihan.catatan',
            'dbo.pelatihan.id_mhs',
            'ref.jenis_pelatihan.nama as jenis_pelatihan',
            'ref.skala_pelatihan.nama as skala_pelatihan',
            DB::raw('DATEDIFF(DAY, dbo.pelatihan.tgl_mulai,
             dbo.pelatihan.tgl_selesai) as rentang_waktu') )
        ->first();

      return $kueriPelatihan;
   }

   public function updateKegiatan(string $idPelatihan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
   {
```

```
        $kueriPelatihan = $this->getForUpdate($idPelatihan, $idMhs, $periode, $idSkem);
        if ($kueriPelatihan == null) {
            throw new AktivitasIsNullException('kegiatan pelatihan');
        }
        $idkegiatanSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this->formatSkala($kueriPelatihan->id_skala_pelatihan), self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $posisi = $this->getKriteria(self::ID_ASPEK_POSISI, $this->formatPosisi(), self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriPelatihan->rentang_waktu), self::ID_BIDANG_SKEM, $idkegiatanSkem);
        $kumpulanElemenPenilaian = [$skala, $posisi, $rentangWaktu];

        $kegiatan = Kegiatan::make(
            $idkegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriPelatihan->nama,
            $kueriPelatihan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );
        return $kegiatan;
    }

    public function updateKumpulanKegiatan(array $kumpulanIdPelatihan, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanKegiatan = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $kegiatan = $this->updateKegiatan($idPelatihan, $idMhs, $idSkem, $tglKlaim, $periode);
```

```
            array_push($kumpulanKegiatan, $kegiatan);
        }
        return $kumpulanKegiatan;
    }

}
```

5.2.3.8 MssqlKompetisiRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use App\Modules\Skem\Domain\Exception\ObjectNotFoundException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\KompetisiRepository;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Kompetisi\Kompetisi;
use Illuminate\Support\Facades\DB;

class MssqlKompetisiRepository implements KompetisiRepository
{
    const DURASI_PORTOFOLIO = 365;
    const ID_JENIS_PORTOFOLIO = 'kompetisi';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 2;

    const ID_ASPEK_JUMLAH_ANGGOTA = 1;
    const ID_ASPEK_SKALA = 2;
    const ID_ASPEK_LUARAN = 3;
```

```
    const ID_ASPEK_RENTANG_WAKTU = 4;
    const ID_ASPEK_BIDANG_ILMU = 5;
    const ID_ASPEK_LEVEL = 6;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kumpulanKompetisi = DB::table('dbo.kompetisi')
            ->join('dbo.anggota_kompetisi', 'dbo.kompetisi.id_kompetisi', '=',
'dbo.anggota_kompetisi.id_kompetisi')
            ->where('dbo.anggota_kompetisi.id_mhs', '=', $idMhs )
            ->where('dbo.kompetisi.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.kompetisi.id_kompetisi', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',        'dbo.kegiatan_skem.id_skem',        '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',            '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',                 '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.kompetisi.nama', 'dbo.kompetisi.id_kompetisi')
            ->get();
        return $kumpulanKompetisi;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }
```

```
   public function get(string $idKompetisi, MahasiswaId $idMhs, int
$periode)
   {
      $idMhs = $idMhs->id();

      $kompetisi = DB::table('dbo.kompetisi')
         ->join('dbo.anggota_kompetisi', 'dbo.kompetisi.id_kompetisi', '=',
'dbo.anggota_kompetisi.id_kompetisi')
         ->join('ref.jenis_kompetisi', 'dbo.kompetisi.id_jenis_kompetisi',
'=', 'ref.jenis_kompetisi.id_jenis_kompetisi')
         ->join('ref.capaian_kompetisi',
'dbo.kompetisi.id_capaian_kompetisi', '=',
'ref.capaian_kompetisi.id_capaian_kompetisi')
         ->join('ref.skala_kompetisi', 'dbo.kompetisi.id_skala_kompetisi',
'=', 'ref.skala_kompetisi.id_skala_kompetisi')
         ->where('dbo.anggota_kompetisi.id_mhs', '=', $idMhs )
         ->where('dbo.kompetisi.id_kompetisi', '=', $idKompetisi )
         ->where('dbo.kompetisi.status_validasi', '=',
self::STATUS_VALIDASI)
         ->whereNotIn('dbo.kompetisi.id_kompetisi', function($query) use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
         })
         ->whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,
GETDATE()) <= ' . $periode )
         ->select('dbo.kompetisi.id_kompetisi',
            'dbo.kompetisi.id_mhs as ketua',
            'dbo.kompetisi.nama',
            'dbo.kompetisi.url_kompetisi',
```

```
            'dbo.kompetisi.tgl_mulai',
            'dbo.kompetisi.tgl_selesai',
            'dbo.kompetisi.luaran',
            'dbo.kompetisi.jenis_luaran',
            'dbo.kompetisi.penyelenggara',
            'dbo.kompetisi.is_regu',
            'dbo.kompetisi.jml_anggota',
            'dbo.kompetisi.is_bidang_ilmu_berhubungan',
            'dbo.kompetisi.jml_peserta',
            'dbo.kompetisi.jml_prov_negara_ikutserta',
            'dbo.kompetisi.id_jenis_kompetisi',
            'dbo.kompetisi.id_capaian_kompetisi',
            'dbo.kompetisi.id_skala_kompetisi',
            'dbo.kompetisi.status_validasi',
            'dbo.kompetisi.tgl_validasi',
            'dbo.kompetisi.id_validator',
            'dbo.kompetisi.nama_validator',
            'dbo.kompetisi.catatan',
            'dbo.anggota_kompetisi.id_mhs',
            'ref.jenis_kompetisi.nama as jenis_kompetisi',
            'ref.skala_kompetisi.nama as skala_kompetisi',
            'ref.capaian_kompetisi.nama as capaian_kompetisi',
            DB::raw('DATEDIFF(MONTH, dbo.kompetisi.tgl_mulai,
             dbo.kompetisi.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kompetisi;
}

public function formatJumlahAnggota(int $jumlahAnggota)
{
    if($jumlahAnggota == 1 ) {
        return 1;
    } elseif ($jumlahAnggota == 2) {
        return 2;
    } elseif ($jumlahAnggota > 2) {
        return 3;
    } else {
```

```
            throw new PadananBobotNotFoundException("jumlah anggota",
'kompetisi');
        }
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu < 3 && $rentangWaktu > -1) {
            return 10;
        } elseif ($rentangWaktu <= 6) {
            return 11;
        } elseif ($rentangWaktu > 6) {
            return 12;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'kompetisi');
        }
    }

    public function formatBidangIlmu(int $bidangIlmu)
    {
        if ($bidangIlmu > 0) {
            return 14;
        } else {
            return 13;
        }
    }

    public function formatLevel(int $idLevel)
    {
        $level = DB::table('ref.jenis_kompetisi')
            ->where('ref.jenis_kompetisi.id_jenis_kompetisi', '=', $idLevel)-
>exists();
        if ($level) {
            return 16;
        } else {
            return 15;
```

```
        }
    }

    public function formatSkala(int $idSkala)
    {
        if ($idSkala == 1) {
            return 4;
        } elseif ($idSkala == 2 || $idSkala == 3) {
            return 5;
        } elseif ($idSkala == 4) {
            return 6;
        } else {
            throw new PadananBobotNotFoundException('skala', 'kompetisi');
        }
    }

    public function formatLuaran(string $luaran)
    {
        if ($luaran == 'A') {
            return 7;
        } elseif ($luaran == 'B') {
            return 8;
        } elseif ($luaran == 'C') {
            return 9;
        } else {
            throw new PadananBobotNotFoundException('luaran',
'kompetisi');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
```

```
        ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
        ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
          'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
          'ref.kriteria_skem.*'
        )
        ->first();

      $kriteria = ElemenPenilaian::make(
        $kueriKriteria->id_kriteria_skem,
        $kueriKriteria->id_aspek_nilai,
        $kueriKriteria->aspek_nilai,
        $kueriKriteria->nama,
        $kueriKriteria->bobot
      );

      return $kriteria;
   }

   public function buildKompetisi(string $idKompetisi, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
   {
      $kueriKompetisi = $this->get($idKompetisi, $idMhs, $periode);
      if ($kueriKompetisi == null) {
        throw new AktivitasIsNullException('kompetisi');
      }
      $idKegiatanSkem = new AktivitasId();

      $jumlahAnggota = $this-
>getKriteria(self::ID_ASPEK_JUMLAH_ANGGOTA, $this-
>formatJumlahAnggota($kueriKompetisi->jml_anggota),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriKompetisi->id_skala_kompetisi),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $luaran = $this->getKriteria(self::ID_ASPEK_LUARAN, $this->formatLuaran($kueriKompetisi->jenis_luaran), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriKompetisi->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $bidangIlmu = $this->getKriteria(self::ID_ASPEK_BIDANG_ILMU, $this->formatBidangIlmu($kueriKompetisi->is_bidang_ilmu_berhubungan), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $level =$this->getKriteria(self::ID_ASPEK_LEVEL, $this->formatlevel($kueriKompetisi->id_jenis_kompetisi), self::ID_BIDANG_SKEM, $idKegiatanSkem);

        $kumpulanElemenPenilaian = [$jumlahAnggota, $skala, $luaran, $rentangWaktu, $bidangIlmu, $level];
        $kompetisi = Kompetisi::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriKompetisi->nama,
            $kueriKompetisi->id_kompetisi,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriKompetisi->id_capaian_kompetisi
        );

        return $kompetisi;
    }

    public function buildKumpulanKompetisi(array $kumpulanIdKompetisi, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        if ($kumpulanIdKompetisi == null || count($kumpulanIdKompetisi) < 1) {
            throw new ObjectNotFoundException("kompetisi");
        }
```

```
        $kumpulanKompetisi = [];
        foreach ($kumpulanIdKompetisi as $idKompetisi) {
            $kompetisi = $this->buildKompetisi($idKompetisi, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanKompetisi, $kompetisi);
        }
        return $kumpulanKompetisi;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanKompetisi = DB::table('dbo.kompetisi')
            ->join('dbo.anggota_kompetisi', 'dbo.kompetisi.id_kompetisi', '=',
'dbo.anggota_kompetisi.id_kompetisi')
            ->where('dbo.anggota_kompetisi.id_mhs', '=', $idMhs )
            ->where('dbo.kompetisi.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.kompetisi.id_kompetisi',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
```

```
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
                })
                  -
>whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,  GETDATE())
<= ' . $periode );
            })
               ->orWhere(function($query3) use ($idMhs, $idSkem) {
                  $query3->whereIn('dbo.kompetisi.id_kompetisi',
function($query4) use ($idMhs, $idSkem) {
                     $query4->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                  });
//                ->whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,
GETDATE()) <= ' . $periode );
               });
         })
         ->select('dbo.kompetisi.nama', 'dbo.kompetisi.id_kompetisi')
         ->get();
      return $kueriKumpulanKompetisi;
   }


   public function getForUpdate(string $idKompetisi, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
   {
      $idMhs = $idMhs->id();
      $idSkem = $idSkem->id();

      $kueriKompetisi = DB::table('dbo.kompetisi')
```

```
        ->join('dbo.anggota_kompetisi', 'dbo.kompetisi.id_kompetisi', '=',
'dbo.anggota_kompetisi.id_kompetisi')
        ->join('ref.jenis_kompetisi',    'dbo.kompetisi.id_jenis_kompetisi',
'=', 'ref.jenis_kompetisi.id_jenis_kompetisi')
        ->join('ref.capaian_kompetisi',
'dbo.kompetisi.id_capaian_kompetisi',                              '=',
'ref.capaian_kompetisi.id_capaian_kompetisi')
        ->join('ref.skala_kompetisi',   'dbo.kompetisi.id_skala_kompetisi',
'=', 'ref.skala_kompetisi.id_skala_kompetisi')
        ->where('dbo.anggota_kompetisi.id_mhs', '=', $idMhs )
        ->where('dbo.kompetisi.id_kompetisi', '=', $idKompetisi )
        ->where('dbo.kompetisi.status_validasi',                           '=',
self::STATUS_VALIDASI)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
           $query0->where(function($query1) use ($idMhs, $periode) {
              $query1->whereNotIn('dbo.kompetisi.id_kompetisi',
function($query2) use ($idMhs) {
                 $query2->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
              })
                 -
>whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,  GETDATE())
<= ' . $periode );
           })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                 $query3->whereIn('dbo.kompetisi.id_kompetisi',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                       ->from('dbo.kegiatan_skem')
```

```
                        ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                  ->whereRaw('DATEDIFF(DAY,dbo.kompetisi.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.kompetisi.id_kompetisi',
            'dbo.kompetisi.id_mhs as ketua',
            'dbo.kompetisi.nama',
            'dbo.kompetisi.url_kompetisi',
            'dbo.kompetisi.tgl_mulai',
            'dbo.kompetisi.tgl_selesai',
            'dbo.kompetisi.luaran',
            'dbo.kompetisi.jenis_luaran',
            'dbo.kompetisi.penyelenggara',
            'dbo.kompetisi.is_regu',
            'dbo.kompetisi.jml_anggota',
            'dbo.kompetisi.is_bidang_ilmu_berhubungan',
            'dbo.kompetisi.jml_peserta',
            'dbo.kompetisi.jml_prov_negara_ikutserta',
            'dbo.kompetisi.id_jenis_kompetisi',
            'dbo.kompetisi.id_capaian_kompetisi',
            'dbo.kompetisi.id_skala_kompetisi',
            'dbo.kompetisi.status_validasi',
            'dbo.kompetisi.tgl_validasi',
            'dbo.kompetisi.id_validator',
            'dbo.kompetisi.nama_validator',
            'dbo.kompetisi.catatan',
            'dbo.anggota_kompetisi.id_mhs',
            'ref.jenis_kompetisi.nama as jenis_kompetisi',
            'ref.skala_kompetisi.nama as skala_kompetisi',
            'ref.capaian_kompetisi.nama as capaian_kompetisi',
```

```
            DB::raw('DATEDIFF(MONTH, dbo.kompetisi.tgl_mulai,
             dbo.kompetisi.tgl_selesai) as rentang_waktu') )
          ->first();

      return $kueriKompetisi;
   }

   public function updateKompetisi(string $idKompetisi, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
   {
      $kueriKompetisi = $this->getForUpdate($idKompetisi, $idMhs,
$periode, $idSkem);
      if ($kueriKompetisi == null) {
         throw new AktivitasIsNullException('kompetisi');
      }
      $idKegiatanSkem = new AktivitasId();

      $jumlahAnggota = $this-
>getKriteria(self::ID_ASPEK_JUMLAH_ANGGOTA, $this-
>formatJumlahAnggota($kueriKompetisi->jml_anggota),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriKompetisi->id_skala_kompetisi),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $luaran = $this->getKriteria(self::ID_ASPEK_LUARAN, $this-
>formatLuaran($kueriKompetisi->jenis_luaran),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriKompetisi->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $bidangIlmu = $this-
>getKriteria(self::ID_ASPEK_BIDANG_ILMU, $this-
>formatBidangIlmu($kueriKompetisi->is_bidang_ilmu_berhubungan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $level =$this->getKriteria(self::ID_ASPEK_LEVEL, $this->formatlevel($kueriKompetisi->id_jenis_kompetisi), self::ID_BIDANG_SKEM, $idKegiatanSkem);

        $kumpulanElemenPenilaian = [$jumlahAnggota, $skala, $luaran, $rentangWaktu, $bidangIlmu, $level];
        $kompetisi = Kompetisi::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriKompetisi->nama,
            $kueriKompetisi->id_kompetisi,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriKompetisi->id_capaian_kompetisi
        );

        return $kompetisi;
    }

    public function updateKumpulanKompetisi(array $kumpulanIdKompetisi, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        if ($kumpulanIdKompetisi == null || count($kumpulanIdKompetisi) < 1) {
            throw new ObjectNotFoundException("kompetisi");
        }
        $kumpulanKompetisi = [];
        foreach ($kumpulanIdKompetisi as $idKompetisi) {
            $kompetisi = $this->updateKompetisi($idKompetisi, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanKompetisi, $kompetisi);
        }
        return $kumpulanKompetisi;
    }

}
```

5.2.3.9 MssqlLkmmPemanduRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use
App\Modules\Skem\Domain\Model\LkmmPemandu\LkmmPemandu;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use
App\Modules\Skem\Domain\Repositories\LkmmPemanduRepository;

use Illuminate\Support\Facades\DB;

class MssqlLkmmPemanduRepository implements
LkmmPemanduRepository
{
    const ID_JENIS_PORTOFOLIO = 'kegiatan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 7;

    const ID_ASPEK_JENIS_PELATIHAN = 28;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
```

```
        ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [5,6,7,8])
        ->where('dbo.kegiatan.status_validasi', '=',
self::STATUS_VALIDASI)
        ->whereNotIn('dbo.kegiatan.id_kegiatan', function($query) use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
        })
        ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode )
        ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
        ->get();
    return $kueriKumpulanKegiatan;
  }

  public function countRequestList(MahasiswaId $idMhs, int $periode)
  {
    $requestList = $this->requestList($idMhs, $periode);
    return count($requestList);
  }

  public function get(string $idLkmmPemandu, MahasiswaId $idMhs, int
$periode)
  {
    $idMhs = $idMhs->id();

    $kegiatan = DB::table('dbo.kegiatan')
        ->join('ref.jenis_kegiatan', 'dbo.kegiatan.id_jenis_kegiatan', '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
        ->join('ref.peran_kegiatan', 'dbo.kegiatan.id_peran_kegiatan', '=',
'ref.peran_kegiatan.id_peran_kegiatan')
```

```
        ->join('ref.skala_kegiatan', 'dbo.kegiatan.id_skala_kegiatan', '=',
'ref.skala_kegiatan.id_skala_kegiatan')
        ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [5,6,7,8])
        ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
        ->where('dbo.kegiatan.id_kegiatan', '=', $idLkmmPemandu )
        ->where('dbo.kegiatan.status_validasi', '=',
self::STATUS_VALIDASI)
        ->whereNotIn('dbo.kegiatan.id_kegiatan', function($query) use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
        })
        ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode)
        ->select('dbo.kegiatan.id_kegiatan',
            'dbo.kegiatan.nama',
            'dbo.kegiatan.tgl_mulai',
            'dbo.kegiatan.tgl_selesai',
            'dbo.kegiatan.deskripsi',
            'dbo.kegiatan.lokasi',
            'dbo.kegiatan.is_bidang_ilmu_berhubungan',
            'dbo.kegiatan.id_jenis_kegiatan',
            'dbo.kegiatan.id_peran_kegiatan',
            'dbo.kegiatan.id_skala_kegiatan',
            'dbo.kegiatan.status_validasi',
            'dbo.kegiatan.tgl_validasi',
            'dbo.kegiatan.id_validator',
            'dbo.kegiatan.nama_validator',
            'dbo.kegiatan.catatan',
            'dbo.kegiatan.id_mhs',
```

```
            'ref.jenis_kegiatan.nama as jenis_kegiatan',
            'ref.skala_kegiatan.nama as skala_kegiatan',
            'ref.peran_kegiatan.nama as posisi_kegiatan',
            DB::raw('DATEDIFF(DAY, dbo.kegiatan.tgl_mulai,
             dbo.kegiatan.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kegiatan;
  }

  public function formatJenisKegiatan(int $idJenisKegiatan)
  {
    switch ($idJenisKegiatan) {
      case 5: //LKMM PRA TD
        return 21;
        break;
      case 6: // LKMM TD
        return 22;
        break;
      case 7: // LKMM TM
        return 23;
        break;
      case 8: // LKMM TL
        return 24;
        break;
      default:
        throw new PadananBobotNotFoundException('jenis kegiatan',
'lkmm pemandu');
    }
  }

  public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
  {
    $kueriKriteria = DB::table('ref.kriteria_skem')
      ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
      ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
      ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
```

```
        ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
        ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
          'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
          'ref.kriteria_skem.*'
        )
        ->first();

      $kriteria = ElemenPenilaian::make(
        $kueriKriteria->id_kriteria_skem,
        $kueriKriteria->id_aspek_nilai,
        $kueriKriteria->aspek_nilai,
        $kueriKriteria->nama,
        $kueriKriteria->bobot
      );

      return $kriteria;
   }

   public function buildLkmmPemandu(string $idLkmmPemandu,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
   {
      $kueriLkmmPemandu = $this->get($idLkmmPemandu, $idMhs,
$periode);
      if ($kueriLkmmPemandu == null) {
        throw new AktivitasIsNullException('lkmm pemandu');
      }
      $idKegiatanSkem = new AktivitasId();
      $jenisKegiatan = $this-
>getKriteria(self::ID_ASPEK_JENIS_PELATIHAN, $this-
>formatJenisKegiatan($kueriLkmmPemandu->id_jenis_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $kumpulanElemenPenilaian = [$jenisKegiatan];

      $lkmmPemandu = LkmmPemandu::make(
        $idKegiatanSkem,
        self::ID_BIDANG_SKEM,
```

```
            $tglKlaim,
            $kueriLkmmPemandu->nama,
            $kueriLkmmPemandu->id_kegiatan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );

        return $lkmmPemandu;
    }

    public function buildKumpulanLkmmPemandu( array $kumpulanIdLkmmPemandu, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanLkmmPemandu = [];
        foreach ($kumpulanIdLkmmPemandu as $idLkmmPemandu) {
            $lkmmPemandu = $this->buildLkmmPemandu($idLkmmPemandu, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanLkmmPemandu, $lkmmPemandu);
        }

        return $kumpulanLkmmPemandu;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate( MahasiswaId $idMhs, int $periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanKegiatan = DB::table('dbo.kegiatan')
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs)
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [5,6,7,8])
```

```
        ->where('dbo.kegiatan.status_validasi',                    '=',
self::STATUS_VALIDASI)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
           $query0->where(function($query1) use ($idMhs, $periode) {
              $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                 $query2->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
              })
                 ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
           })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                 $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                       ->from('dbo.kegiatan_skem')
                       ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                       ->where('dbo.skem.id_mhs', '=', $idMhs)
                       ->where('dbo.skem.id_skem', '=', $idSkem)
                       ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                       ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                 });
//                  ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
              });
        })
```

```
            ->select('dbo.kegiatan.nama', 'dbo.kegiatan.id_kegiatan')
            ->get();
        return $kueriKumpulanKegiatan;
    }

    public function getForUpdate(string $idLkmmPemandu, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKegiatan = DB::table('dbo.kegiatan')
            ->join('ref.jenis_kegiatan',   'dbo.kegiatan.id_jenis_kegiatan',   '=',
'ref.jenis_kegiatan.id_jenis_kegiatan')
            ->join('ref.peran_kegiatan',  'dbo.kegiatan.id_peran_kegiatan',  '=',
'ref.peran_kegiatan.id_peran_kegiatan')
            ->join('ref.skala_kegiatan',   'dbo.kegiatan.id_skala_kegiatan',   '=',
'ref.skala_kegiatan.id_skala_kegiatan')
            ->whereIn('dbo.kegiatan.id_jenis_kegiatan', [5,6,7,8])
            ->where('dbo.kegiatan.id_mhs', '=', $idMhs )
            ->where('dbo.kegiatan.id_kegiatan', '=', $idLkmmPemandu )
            ->where('dbo.kegiatan.status_validasi',                              '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.kegiatan.id_kegiatan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                    ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
```

```
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.kegiatan.id_kegiatan',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                   ->whereRaw('DATEDIFF(DAY,dbo.kegiatan.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.kegiatan.id_kegiatan',
            'dbo.kegiatan.nama',
            'dbo.kegiatan.tgl_mulai',
            'dbo.kegiatan.tgl_selesai',
            'dbo.kegiatan.deskripsi',
            'dbo.kegiatan.lokasi',
            'dbo.kegiatan.is_bidang_ilmu_berhubungan',
            'dbo.kegiatan.id_jenis_kegiatan',
            'dbo.kegiatan.id_peran_kegiatan',
            'dbo.kegiatan.id_skala_kegiatan',
            'dbo.kegiatan.status_validasi',
            'dbo.kegiatan.tgl_validasi',
            'dbo.kegiatan.id_validator',
            'dbo.kegiatan.nama_validator',
            'dbo.kegiatan.catatan',
            'dbo.kegiatan.id_mhs',
            'ref.jenis_kegiatan.nama as jenis_kegiatan',
            'ref.skala_kegiatan.nama as skala_kegiatan',
```

```
            'ref.peran_kegiatan.nama as posisi_kegiatan',
            DB::raw('DATEDIFF(DAY, dbo.kegiatan.tgl_mulai,
             dbo.kegiatan.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriKegiatan;
  }

  public function updateLkmmPemandu(string $idLkmmPemandu,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
  {
    $kueriLkmmPemandu = $this->getForUpdate($idLkmmPemandu,
$idMhs, $periode, $idSkem);
    if ($kueriLkmmPemandu == null) {
      throw new AktivitasIsNullException('lkmm pemandu');
    }
    $idKegiatanSkem = new AktivitasId();
    $jenisKegiatan = $this-
>getKriteria(self::ID_ASPEK_JENIS_PELATIHAN, $this-
>formatJenisKegiatan($kueriLkmmPemandu->id_jenis_kegiatan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
    $kumpulanElemenPenilaian = [$jenisKegiatan];

    $lkmmPemandu = LkmmPemandu::make(
      $idKegiatanSkem,
      self::ID_BIDANG_SKEM,
      $tglKlaim,
      $kueriLkmmPemandu->nama,
      $kueriLkmmPemandu->id_kegiatan,
      self::ID_JENIS_PORTOFOLIO,
      $kumpulanElemenPenilaian,
      '4'
    );

    return $lkmmPemandu;
  }
```

```
    public function updateKumpulanLkmmPemandu( array $kumpulanIdLkmmPemandu, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanLkmmPemandu = [];
        foreach ($kumpulanIdLkmmPemandu as $idLkmmPemandu) {
            $lkmmPemandu = $this->updateLkmmPemandu($idLkmmPemandu, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanLkmmPemandu, $lkmmPemandu);
        }

        return $kumpulanLkmmPemandu;
    }
}
```

5.2.3.10 MssqlMagangRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use App\Modules\Skem\Domain\Exception\PadananBobotNotFoundException;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Magang\Magang;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\MagangRepository;

use Illuminate\Support\Facades\DB;

class MssqlMagangRepository implements MagangRepository
```

```
{
    const ID_JENIS_PORTOFOLIO = 'magang';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 4;
    const DURASI_MINIMUM_MAGANG = 3; //Dalam Bulan

    const ID_ASPEK_TEMPAT_MAGANG = 11;
//    const ID_BOBOT_PERGURUAN_TINGGI = 35;
//    const ID_BOBOT_LEMBAGA_PENELITIAN = 36;
//    const ID_BOBOT_PEMERINTAH = 37;
//    const ID_BOBOT_PERUSAHAAN_SWASTA = 38;
//    const ID_BOBOT_BUMN = 39;

    const ID_ASPEK_SKALA = 12;
//    const ID_BOBOT_INSTITUT = 40;
//    const ID_BOBOT_NASIONAL = 41;
//    const ID_BOBOT_INTERNASIONAL = 42;

    const ID_ASPEK_RENTANG_WAKTU = 13;
//    const ID_BOBOT_DIBAWAH_TIGA_BULAN = 43;
//    const ID_BOBOT_TIGA_SAMPAI_EMPAT_BULAN = 44;
//    const ID_BOBOT_DIATAS_LIMA_BULAN = 45;

    const ID_ASPEK_BIDANG_ILMU = 14;
//    const ID_BOBOT_TIDAK_BERHUBUNGAN = 46;
//    const ID_BOBOT_BERHUBUNGAN = 47;

    public function requestList( MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi',          'dbo.magang.id_jenis_dudi',          '=',
'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang',    'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang',    'dbo.magang.id_skala_magang',    '=',
'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
```

```
        ->where('dbo.magang.status_validasi', '=',
self::STATUS_VALIDASI)
        ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
        ->whereNotIn('dbo.magang.id_magang', function($query) use
($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
        })
        ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode)
        ->whereRaw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
dbo.magang.tgl_selesai) >= ' . self::DURASI_MINIMUM_MAGANG)

        ->select('dbo.magang.id_magang', 'dbo.magang.tempat',
'dbo.magang.deskripsi_kerja')
        ->get();
    return $kueriKumpulanMagang;
  }

  public function countRequestList(MahasiswaId $idMhs, int $periode)
  {
    $requestList = $this->requestList($idMhs, $periode);
    return count($requestList);
  }

  public function get( string $idMagang, MahasiswaId $idMhs, int
$periode)
  {
    $idMhs = $idMhs->id();
```

```
        $kueriMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi',          'dbo.magang.id_jenis_dudi',          '=',
'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang',   'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang',    'dbo.magang.id_skala_magang',    '=',
'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
            ->where('dbo.magang.id_magang', '=', $idMagang)
            ->where('dbo.magang.status_validasi',                                '=',
self::STATUS_VALIDASI)
            ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
            ->whereNotIn('dbo.magang.id_magang',    function($query)    use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',        'dbo.kegiatan_skem.id_skem',        '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',             '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',                  '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->whereRaw('DATEDIFF(MONTH,dbo.magang.tgl_mulai,
dbo.magang.tgl_selesai) >= ' . self::DURASI_MINIMUM_MAGANG)
            ->select('dbo.magang.*',
                'ref.jenis_dudi.nama as tempat_magang',
                'ref.kategori_magang.nama as kategori',
                'ref.skala_magang.nama as skala',
                DB::raw('DATEDIFF(MONTH,            dbo.magang.tgl_mulai,
ISNULL(dbo.magang.tgl_selesai, GETDATE())) as rentang_waktu'))
            ->first();

        return $kueriMagang;
    }
```

```
    public function formatTempatMagang(int $idTempatMagang)
    {
        if ($idTempatMagang == 1) {
            return 35; // perguruan tinggi
        } elseif ($idTempatMagang == 2) {
            return 36; // lembaga penelitian
        } elseif ($idTempatMagang == 3) {
            return 37; //pemerintah
        } elseif ($idTempatMagang == 4) {
            return 38; // perusahaan swasta
        } else if ($idTempatMagang == 5) {
            return 39; // BUMN
        } else {
            throw new PadananBobotNotFoundException('tempat magang',
'magang');
        }
    }

    public function formatSkala (int $idSkala)
    {
        if ($idSkala == 1) {
            return 40; //institut
        } elseif ($idSkala == 2) {
            return 41; // nasional
        } elseif ($idSkala == 3) {
            return 42; // internasional
        } else {
            throw new PadananBobotNotFoundException('skala', 'magang');
        }
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu < 3 && $rentangWaktu > -1) {
            return 43;
        } elseif ($rentangWaktu < 5) {
            return 44;
```

```
        } elseif ($rentangWaktu >= 5) {
            return 45;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'magang');
        }
    }

    public function formatBidangIlmu(int $bidangIlmu)
    {
        if ($bidangIlmu > 0) {
            return 47;
        } else {
            return 46;
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
```

```
        );

        return $kriteria;
    }

    public function buildMagang(string $idMagang, MahasiswaId $idMhs,
SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriMagang = $this->get($idMagang, $idMhs, $periode);
        if ($kueriMagang == null) {
            throw new AktivitasIsNullException('magang');
        }
        $idKegiatanSkem = new AktivitasId();
        $tempatMagang                    =                    $this-
>getKriteria(self::ID_ASPEK_TEMPAT_MAGANG,                    $this-
>formatTempatMagang($kueriMagang->id_jenis_dudi),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $skala    =    $this->getKriteria(self::ID_ASPEK_SKALA,    $this-
>formatSkala($kueriMagang->id_skala_magang),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu                    =                    $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU,                    $this-
>formatRentangWaktu($kueriMagang->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $bidangIlmu                      =                    $this-
>getKriteria(self::ID_ASPEK_BIDANG_ILMU,                      $this-
>formatBidangIlmu($kueriMagang->is_bidang_ilmu_berhubungan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian    =    [$tempatMagang,    $skala,
$rentangWaktu, $bidangIlmu];

        $magang = Magang::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriMagang->deskripsi_kerja,
            $kueriMagang->id_magang,
```

```
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriMagang->id_kategori_magang
        );

        return $magang;
    }

    public function buildKumpulanMagang(array $kumpulanIdMagang, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanMagang = [];
        foreach ($kumpulanIdMagang as $idMagang) {
            $magang = $this->buildMagang($idMagang, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanMagang, $magang);
        }

        return $kumpulanMagang;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int $periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanMagang = DB::table('dbo.magang')
            ->join('ref.jenis_dudi', 'dbo.magang.id_jenis_dudi', '=', 'ref.jenis_dudi.id_jenis_dudi')
            ->join('ref.kategori_magang', 'dbo.magang.id_kategori_magang', '=', 'ref.kategori_magang.id_kategori_magang')
            ->join('ref.skala_magang', 'dbo.magang.id_skala_magang', '=', 'ref.skala_magang.id_skala_magang')
            ->where('dbo.magang.id_mhs', '=', $idMhs)
```

```
        ->where('dbo.magang.status_validasi',                    '=',
self::STATUS_VALIDASI)
        ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
        ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
           $query0->where(function($query1) use ($idMhs, $periode) {
              $query1->whereNotIn('dbo.magang.id_magang',
function($query2) use ($idMhs) {
                 $query2->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
              })
                 ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode )
                 -
>whereRaw('DATEDIFF(MONTH,dbo.magang.tgl_mulai,
dbo.magang.tgl_selesai) >= ' . self::DURASI_MINIMUM_MAGANG);
           })
              ->orWhere(function($query3) use ($idMhs, $idSkem) {
                 $query3->whereIn('dbo.magang.id_magang',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                       ->from('dbo.kegiatan_skem')
                       ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                       ->where('dbo.skem.id_mhs', '=', $idMhs)
                       ->where('dbo.skem.id_skem', '=', $idSkem)
                       ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                       ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                 });
```

```
//                  ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.magang.id_magang',              'dbo.magang.tempat',
'dbo.magang.deskripsi_kerja')
        ->get();

    return $kueriKumpulanMagang;
  }

  public function getForUpdate(string $idMagang, MahasiswaId $idMhs,
int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriMagang = DB::table('dbo.magang')
      ->join('ref.jenis_dudi',        'dbo.magang.id_jenis_dudi',        '=',
'ref.jenis_dudi.id_jenis_dudi')
      ->join('ref.kategori_magang',  'dbo.magang.id_kategori_magang',
'=', 'ref.kategori_magang.id_kategori_magang')
      ->join('ref.skala_magang',   'dbo.magang.id_skala_magang',   '=',
'ref.skala_magang.id_skala_magang')
      ->where('dbo.magang.id_mhs', '=', $idMhs)
      ->where('dbo.magang.id_magang', '=', $idMagang)
      ->where('dbo.magang.status_validasi',                          '=',
self::STATUS_VALIDASI)
      ->where('dbo.magang.is_magang_konversi_akademik', '=', 0)
      ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
        $query0->where(function($query1) use ($idMhs, $periode) {
          $query1->whereNotIn('dbo.magang.id_magang',
function($query2) use ($idMhs) {
            $query2->select('dbo.kegiatan_skem.id_portofolio')
              ->from('dbo.kegiatan_skem')
              ->join('dbo.skem',   'dbo.kegiatan_skem.id_skem',   '=',
'dbo.skem.id_skem')
              ->where('dbo.skem.id_mhs', '=', $idMhs)
```

```
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                })
                ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode )
                ->whereRaw('DATEDIFF(MONTH,
dbo.magang.tgl_mulai, dbo.magang.tgl_selesai) >= ' .
self::DURASI_MINIMUM_MAGANG);
            })
            ->orWhere(function($query3) use ($idMhs, $idSkem) {
                $query3->whereIn('dbo.magang.id_magang',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                    });
//                  ->whereRaw('DATEDIFF(DAY,dbo.magang.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.magang.*',
            'ref.jenis_dudi.nama as tempat_magang',
            'ref.kategori_magang.nama as kategori',
            'ref.skala_magang.nama as skala',
            DB::raw('DATEDIFF(MONTH, dbo.magang.tgl_mulai,
ISNULL(dbo.magang.tgl_selesai, GETDATE())) as rentang_waktu'))
        ->first();
```

```
        return $kueriMagang;
    }

    public function updateMagang(string $idMagang, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriMagang = $this->getForUpdate($idMagang, $idMhs,
$periode, $idSkem);
        if ($kueriMagang == null) {
            throw new AktivitasIsNullException('magang');
        }
        $idKegiatanSkem = new AktivitasId();
        $tempatMagang = $this-
>getKriteria(self::ID_ASPEK_TEMPAT_MAGANG, $this-
>formatTempatMagang($kueriMagang->id_jenis_dudi),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriMagang->id_skala_magang),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriMagang->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $bidangIlmu = $this-
>getKriteria(self::ID_ASPEK_BIDANG_ILMU, $this-
>formatBidangIlmu($kueriMagang->is_bidang_ilmu_berhubungan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$tempatMagang, $skala,
$rentangWaktu, $bidangIlmu];

        $magang = Magang::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriMagang->deskripsi_kerja,
            $kueriMagang->id_magang,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriMagang->id_kategori_magang
```

```
        );

        return $magang;
    }

    public function updateKumpulanMagang(array $kumpulanIdMagang,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanMagang = [];
        foreach ($kumpulanIdMagang as $idMagang) {
            $magang = $this->updateMagang($idMagang, $idMhs, $idSkem,
$tglKlaim, $periode);
            array_push($kumpulanMagang, $magang);
        }

        return $kumpulanMagang;
    }
}
```

5.2.3.11 MssqlMahasiswaRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Model\Mahasiswa\Mahasiswa;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\Semester;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Validator\ValidatorId;
use App\Modules\Skem\Domain\Repositories\MahasiswaRepository;

use Illuminate\Support\Facades\DB;
use Mockery\Exception;

class MssqlMahasiswaRepository implements MahasiswaRepository
{
```

```
    public function get(MahasiswaId $idMhs)
    {
        $idMahasiswa = $idMhs->id();

        $kueriMahasiswa = DB::table('dbo.mahasiswa')
            ->where('dbo.mahasiswa.id_mhs', '=', $idMahasiswa)
            ->first();

        if($kueriMahasiswa == null) {
            throw new Exception('Fungi get pada Repositori Mahasiswa tidak
menemukan mahasiswa yang dicari');
        }

        $mahasiswa = new Mahasiswa($idMhs, $kueriMahasiswa->nama,
$kueriMahasiswa->thn_angkatan, $kueriMahasiswa->nrp);

        $kueriKumpulanSemester = DB::table('ref.semester')
            ->where('ref.semester.id_thn_ajar', '>=', $mahasiswa-
>getThnAngkatan())
            ->whereRaw('ref.semester.tgl_mulai <= GETDATE()')
            ->oldest('ref.semester.tgl_mulai')
            ->get();

        $counter = 1;
        foreach ($kueriKumpulanSemester as $kueriSemester) {
            $semester = new Semester(
                $mahasiswa->getIdMhs(),
                new SemesterId($kueriSemester->id_smt),
                $counter,
                $kueriSemester->nama
            );
            $mahasiswa->addSemester($semester);
            $counter += 1;
        }

        return $mahasiswa;
    }

    public function getKumpulanAnakWali(ValidatorId $idValidator)
```

```
    {
        $kueriKumpulanAnakWali = DB::table('dbo.mahasiswa')
            ->where('dbo.mahasiswa.id_dosen_wali', '=', $idValidator->id())
            ->get();

        $kumpulanAnakWali = [];
        foreach ($kueriKumpulanAnakWali as $kueriAnakWali)
        {
            $anakWali = $this->get(new MahasiswaId($kueriAnakWali->id_mhs));
            $kumpulanAnakWali[] = $anakWali;
        }

        return $kumpulanAnakWali;
    }
}
```

5.2.3.12 MssqlOrmawaRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use App\Modules\Skem\Domain\Exception\PadananBobotNotFoundException;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Ormawa\Ormawa;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\OrmawaRepository;

use Illuminate\Support\Facades\DB;
```

```
class MssqlOrmawaRepository implements OrmawaRepository
{
   const ID_JENIS_PORTOFOLIO = 'ormawa';
   const STATUS_VALIDASI = 2;
   const ID_BIDANG_SKEM = 6;

   const ID_ASPEK_SKALA = 15;
   const ID_ASPEK_RENTANG_WAKTU = 16;

   public function requestList( MahasiswaId $idMhs, int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriKumpulanOrmawa = DB::table('dbo.ormawa')
         ->join('dbo.anggota_ormawa', 'dbo.ormawa.id_ormawa', '=',
'dbo.anggota_ormawa.id_ormawa')
         ->where('dbo.anggota_ormawa.id_mhs', '=', $idMhs )
         ->where('dbo.anggota_ormawa.status_validasi', '=',
self::STATUS_VALIDASI)
         ->whereNotIn('dbo.anggota_ormawa.id_anggota_ormawa',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
         })
         -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode)
         ->select('dbo.ormawa.nama', 'dbo.ormawa.id_ormawa',
'dbo.anggota_ormawa.id_anggota_ormawa',
DB::raw('YEAR(dbo.anggota_ormawa.tgl_mulai) as thn_mulai'),
DB::raw('YEAR(dbo.anggota_ormawa.tgl_selesai) as thn_selesai'))
         ->get();
```

```
        return $kueriKumpulanOrmawa;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idAnggotaOrmawa,  MahasiswaId  $idMhs,
int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriOrmawa = DB::table('dbo.ormawa')
            ->join('dbo.anggota_ormawa',     'dbo.ormawa.id_ormawa',     '=',
'dbo.anggota_ormawa.id_ormawa')
            ->join('ref.skala_ormawa',   'dbo.ormawa.id_skala_ormawa',   '=',
'ref.skala_ormawa.id_skala_ormawa')
            ->join('ref.peran_anggota_ormawa',
'dbo.anggota_ormawa.id_peran_anggota_ormawa',                          '=',
'ref.peran_anggota_ormawa.id_peran_anggota_ormawa')
            ->where('dbo.anggota_ormawa.id_mhs', '=', $idMhs )
            ->where('dbo.anggota_ormawa.id_anggota_ormawa',            '=',
$idAnggotaOrmawa )
            ->where('dbo.anggota_ormawa.status_validasi',              '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.ormawa.id_ormawa',   function($query)   use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',       'dbo.kegiatan_skem.id_skem',       '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',           '=',
self::ID_JENIS_PORTOFOLIO)
```

```
                ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
            })
            -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.ormawa.id_ormawa',
                'dbo.ormawa.nama',
                'dbo.ormawa.id_jenis_ormawa',
                'dbo.ormawa.id_skala_ormawa',
                'dbo.anggota_ormawa.tgl_mulai',
                'dbo.anggota_ormawa.tgl_selesai',
                'dbo.anggota_ormawa.status_validasi',
                'dbo.anggota_ormawa.tgl_validasi',
                'dbo.anggota_ormawa.id_validator',
                'dbo.anggota_ormawa.nama_validator',
                'dbo.anggota_ormawa.catatan',
                'dbo.anggota_ormawa.id_mhs',
                'dbo.anggota_ormawa.id_anggota_ormawa',
                'dbo.anggota_ormawa.id_peran_anggota_ormawa',
                'ref.skala_ormawa.nama as skala_ormawa',
                'ref.peran_anggota_ormawa.nama as jabatan',
                DB::raw('DATEDIFF(MONTH,
dbo.anggota_ormawa.tgl_mulai,
                 dbo.anggota_ormawa.tgl_selesai) as rentang_waktu') )
            ->first();

        return $kueriOrmawa;
    }

    public function formatSkala(int $idSkala)
    {
        if ($idSkala == 1) {
            return 48; // departemen
        } elseif ($idSkala == 2) {
            return 49; // fakultas
        } elseif ($idSkala == 3) {
            return 50; // institut
        }  else {
```

```
            throw new PadananBobotNotFoundException('skala', 'ormawa');
        }
    }

    public function formatRentangWaktu(int $rentangWaktu)
    {
        if ($rentangWaktu < 12 && $rentangWaktu > -1) {
            return 51;
        } elseif ($rentangWaktu <= 12) {
            return 52;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'ormawa');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
```

```
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildOrmawa(string $idAnggotaOrmawa,
MahasiswaId  $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriOrmawa = $this->get($idAnggotaOrmawa, $idMhs,
$periode);
        if ($kueriOrmawa == null) {
            throw new AktivitasIsNullException('ormawa');
        }
        $idBidangSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriOrmawa->id_skala_ormawa),
self::ID_BIDANG_SKEM, $idBidangSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriOrmawa->rentang_waktu),
self::ID_BIDANG_SKEM, $idBidangSkem);
        $kumpulanElemenPenilaian = [$skala, $rentangWaktu];
        $ormawa = Ormawa::make(
            $idBidangSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim, $kueriOrmawa->nama,
            $kueriOrmawa->id_anggota_ormawa,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriOrmawa->id_peran_anggota_ormawa
        );
        return $ormawa;
    }

    public function buildKumpulanOrmawa($kumpulanIdOrmawa,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
```

```
        $kumpulanOrmawa = [];
        foreach ($kumpulanIdOrmawa as $idAnggotaOrmawa) {
            $ormawa = $this->buildOrmawa($idAnggotaOrmawa, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanOrmawa, $ormawa);
        }
        return $kumpulanOrmawa;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanOrmawa = DB::table('dbo.ormawa')
            ->join('dbo.anggota_ormawa', 'dbo.ormawa.id_ormawa', '=',
'dbo.anggota_ormawa.id_ormawa')
            ->where('dbo.anggota_ormawa.id_mhs', '=', $idMhs )
            ->where('dbo.anggota_ormawa.status_validasi', '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1-
>whereNotIn('dbo.anggota_ormawa.id_anggota_ormawa',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
```

```
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                })
                    -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3-
>whereIn('dbo.anggota_ormawa.id_anggota_ormawa', function($query4)
use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                        });
//                                                                    -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.ormawa.nama', 'dbo.ormawa.id_ormawa',
'dbo.anggota_ormawa.id_anggota_ormawa',
DB::raw('YEAR(dbo.anggota_ormawa.tgl_mulai) as thn_mulai'),
DB::raw('YEAR(dbo.anggota_ormawa.tgl_selesai) as thn_selesai'))
        ->get();
    return $kueriKumpulanOrmawa;
}

public function getForUpdate(string $idAnggotaOrmawa,
MahasiswaId $idMhs, int $periode, SkemId $idSkem)
{
    $idMhs = $idMhs->id();
```

```
        $idSkem = $idSkem->id();

        $kueriOrmawa = DB::table('dbo.ormawa')
            ->join('dbo.anggota_ormawa',      'dbo.ormawa.id_ormawa',      '=',
'dbo.anggota_ormawa.id_ormawa')
            ->join('ref.skala_ormawa',    'dbo.ormawa.id_skala_ormawa',    '=',
'ref.skala_ormawa.id_skala_ormawa')
            ->join('ref.peran_anggota_ormawa',
'dbo.anggota_ormawa.id_peran_anggota_ormawa',                              '=',
'ref.peran_anggota_ormawa.id_peran_anggota_ormawa')
            ->where('dbo.anggota_ormawa.id_mhs', '=', $idMhs )
            ->where('dbo.anggota_ormawa.id_anggota_ormawa',                '=',
$idAnggotaOrmawa )
            ->where('dbo.anggota_ormawa.status_validasi',                  '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1-
>whereNotIn('dbo.anggota_ormawa.id_anggota_ormawa',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode );
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
```

```
                $query3-
>whereIn('dbo.anggota_ormawa.id_anggota_ormawa', function($query4)
use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                });
//                                                                          -
>whereRaw('DATEDIFF(DAY,dbo.anggota_ormawa.tgl_selesai,
GETDATE()) <= ' . $periode );
            });
        })
        ->select('dbo.ormawa.id_ormawa',
            'dbo.ormawa.nama',
            'dbo.ormawa.id_jenis_ormawa',
            'dbo.ormawa.id_skala_ormawa',
            'dbo.anggota_ormawa.tgl_mulai',
            'dbo.anggota_ormawa.tgl_selesai',
            'dbo.anggota_ormawa.status_validasi',
            'dbo.anggota_ormawa.tgl_validasi',
            'dbo.anggota_ormawa.id_validator',
            'dbo.anggota_ormawa.nama_validator',
            'dbo.anggota_ormawa.catatan',
            'dbo.anggota_ormawa.id_mhs',
            'dbo.anggota_ormawa.id_anggota_ormawa',
            'dbo.anggota_ormawa.id_peran_anggota_ormawa',
            'ref.skala_ormawa.nama as skala_ormawa',
            'ref.peran_anggota_ormawa.nama as jabatan',
            DB::raw('DATEDIFF(MONTH,
dbo.anggota_ormawa.tgl_mulai,
             dbo.anggota_ormawa.tgl_selesai) as rentang_waktu') )
        ->first();
```

```
        return $kueriOrmawa;
    }

    public function updateOrmawa(string $idAnggotaOrmawa,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriOrmawa = $this->getForUpdate($idAnggotaOrmawa,
$idMhs, $periode, $idSkem);
        if ($kueriOrmawa == null) {
            throw new AktivitasIsNullException('ormawa');
        }
        $idBidangSkem = new AktivitasId();

        $skala = $this->getKriteria(self::ID_ASPEK_SKALA, $this-
>formatSkala($kueriOrmawa->id_skala_ormawa),
self::ID_BIDANG_SKEM, $idBidangSkem);
        $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriOrmawa->rentang_waktu),
self::ID_BIDANG_SKEM, $idBidangSkem);
        $kumpulanElemenPenilaian = [$skala, $rentangWaktu];
        $ormawa = Ormawa::make(
            $idBidangSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriOrmawa->nama,
            $kueriOrmawa->id_anggota_ormawa,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriOrmawa->id_peran_anggota_ormawa
        );
        return $ormawa;
    }

    public function updateKumpulanOrmawa(array $kumpulanIdOrmawa,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
```

```
    {
        $kumpulanOrmawa = [];
        foreach ($kumpulanIdOrmawa as $idAnggotaOrmawa) {
            $ormawa = $this->updateOrmawa($idAnggotaOrmawa, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanOrmawa, $ormawa);
        }
        return $kumpulanOrmawa;
    }

}
```

5.2.3.13 MssqlPelatihanRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\PadananBobotNotFoundExcepti
on;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Pelatihan\Pelatihan;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\PelatihanRepository;

use Illuminate\Support\Facades\DB;

class  MssqlPelatihanRepository implements PelatihanRepository
{
    const ID_JENIS_PORTOFOLIO = 'pelatihan';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 10;

    const ID_ASPEK_JENIS_PELATIHAN = 27;
```

```
    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih',
'ref.jenis_pelatihan.id_jenis_latih')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi', '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use
($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode)
            ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
            ->get();
        return $kueriKumpulanPelatihan;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }
```

```
   public function get(string $idPelatihan, MahasiswaId $idMhs, int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriPelatihan = DB::table('dbo.pelatihan')
         ->join('ref.jenis_pelatihan', 'dbo.pelatihan.id_jenis_latih', '=', 'ref.jenis_pelatihan.id_jenis_latih')
         ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
         ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
         ->where('dbo.pelatihan.status_validasi', '=', self::STATUS_VALIDASI)
         ->whereNotIn('dbo.pelatihan.id_jenis_latih', [9999])
         ->whereNotIn('dbo.pelatihan.id_pelatihan', function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=', 'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=', self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem', '=', self::ID_BIDANG_SKEM);
         })
         ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai, GETDATE()) <= ' . $periode )
         ->select('dbo.pelatihan.id_pelatihan',
            'dbo.pelatihan.nama',
            'dbo.pelatihan.tgl_mulai',
            'dbo.pelatihan.tgl_selesai',
            'dbo.pelatihan.penyelenggara',
            'dbo.pelatihan.id_jenis_latih',
            'dbo.pelatihan.status_validasi',
            'dbo.pelatihan.tgl_validasi',
            'dbo.pelatihan.id_validator',
            'dbo.pelatihan.nama_validator',
            'dbo.pelatihan.catatan',
            'dbo.pelatihan.id_mhs',
```

```
            'ref.jenis_pelatihan.nama as jenis_pelatihan',
            DB::raw('DATEDIFF(DAY, dbo.pelatihan.tgl_mulai,
             dbo.pelatihan.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriPelatihan;
  }

  public function formatJenisPelatihan(int $idJenisPelatihan)
  {
    switch ($idJenisPelatihan) {
      case 1: //LKMM PRA TD
        return 17;
        break;
      case 2: // LKMM TD
      case 5: // LKMW TD
        return 18;
        break;
      case 3: // LKMM TM
      case 6: // LKMW TM
        return 19;
        break;
      case 4: // LKMM TL
      case 7: // Innovator Bootcamp/Digital Marketing Camp
      case 8: // ITS Entrepreneur Club (IEC)
      case 9: // Investor Meeting and Exhibition
      case 10: // USAID-JAPRI
    return 20;
        break;
      default:
        throw new PadananBobotNotFoundException('jenis pelatihan',
'lkmm pelatihan');
    }
  }
```

```
    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem',                '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildPelatihan(string $idPelatihan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriPelatihan = $this->get($idPelatihan, $idMhs, $periode);
        if ($kueriPelatihan == null) {
            throw new AktivitasIsNullException('pelatihan');
        }
        $idKegiatanSkem = new AktivitasId();
        $jenisPelatihan                =                $this-
>getKriteria(self::ID_ASPEK_JENIS_PELATIHAN,                $this-
>formatJenisPelatihan($kueriPelatihan->id_jenis_latih),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $kumpulanElemenPenilaian = [$jenisPelatihan];

        $pelatihan = Pelatihan::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriPelatihan->nama,
            $kueriPelatihan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
    );

        return $pelatihan;
    }

    public function buildKumpulanPelatihan(array $kumpulanIdPelatihan,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanPelatihan = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $pelatihan = $this->buildPelatihan($idPelatihan, $idMhs,
$idSkem, $tglKlaim, $periode);
            array_push($kumpulanPelatihan, $pelatihan);
        }

        return $kumpulanPelatihan;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
```

```
        $idSkem = $idSkem->id();

        $kueriKumpulanPelatihan = DB::table('dbo.pelatihan')
            ->join('ref.jenis_pelatihan',     'dbo.pelatihan.id_jenis_latih',     '=',
'ref.jenis_pelatihan.id_jenis_latih')
            ->where('dbo.pelatihan.id_mhs', '=', $idMhs)
            ->where('dbo.pelatihan.status_validasi',                              '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.pelatihan.id_jenis_latih', [9999])
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
```

```
                    });
//                  ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.pelatihan.nama', 'dbo.pelatihan.id_pelatihan')
        ->get();
    return $kueriKumpulanPelatihan;
  }

  public function getForUpdate(string $idPelatihan, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
  {
    $idMhs = $idMhs->id();
    $idSkem = $idSkem->id();

    $kueriPelatihan = DB::table('dbo.pelatihan')
      ->join('ref.jenis_pelatihan',    'dbo.pelatihan.id_jenis_latih',    '=',
'ref.jenis_pelatihan.id_jenis_latih')
      ->where('dbo.pelatihan.id_mhs', '=', $idMhs )
      ->where('dbo.pelatihan.id_pelatihan', '=', $idPelatihan )
      ->where('dbo.pelatihan.status_validasi',                    '=',
self::STATUS_VALIDASI)
      ->whereNotIn('dbo.pelatihan.id_jenis_latih', [9999])
      ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
        $query0->where(function($query1) use ($idMhs, $periode) {
          $query1->whereNotIn('dbo.pelatihan.id_pelatihan',
function($query2) use ($idMhs) {
            $query2->select('dbo.kegiatan_skem.id_portofolio')
              ->from('dbo.kegiatan_skem')
              ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
              ->where('dbo.skem.id_mhs', '=', $idMhs)
              ->where('dbo.kegiatan_skem.id_jenis_portofolio',    '=',
self::ID_JENIS_PORTOFOLIO)
              ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
```

```
                })
                    ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                })
                    ->orWhere(function($query3) use ($idMhs, $idSkem) {
                        $query3->whereIn('dbo.pelatihan.id_pelatihan',
function($query4) use ($idMhs, $idSkem) {
                            $query4->select('dbo.kegiatan_skem.id_portofolio')
                                ->from('dbo.kegiatan_skem')
                                ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                                ->where('dbo.skem.id_mhs', '=', $idMhs)
                                ->where('dbo.skem.id_skem', '=', $idSkem)
                                ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                                ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                            });
//                  ->whereRaw('DATEDIFF(DAY,dbo.pelatihan.tgl_selesai,
GETDATE()) <= ' . $periode );
                    });
            })
            ->select('dbo.pelatihan.id_pelatihan',
                'dbo.pelatihan.nama',
                'dbo.pelatihan.tgl_mulai',
                'dbo.pelatihan.tgl_selesai',
                'dbo.pelatihan.penyelenggara',
                'dbo.pelatihan.id_jenis_latih',
                'dbo.pelatihan.status_validasi',
                'dbo.pelatihan.tgl_validasi',
                'dbo.pelatihan.id_validator',
                'dbo.pelatihan.nama_validator',
                'dbo.pelatihan.catatan',
                'dbo.pelatihan.id_mhs',
                'ref.jenis_pelatihan.nama as jenis_pelatihan',
                DB::raw('DATEDIFF(DAY, dbo.pelatihan.tgl_mulai,
                 dbo.pelatihan.tgl_selesai) as rentang_waktu') )
            ->first();
```

```
        return $kueriPelatihan;
    }

    public function updatePelatihan(string $idPelatihan, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim,  int $periode)
    {
        $kueriPelatihan = $this->getForUpdate($idPelatihan, $idMhs,
$periode, $idSkem);
        if ($kueriPelatihan == null) {
            throw new AktivitasIsNullException('pelatihan');
        }
        $idKegiatanSkem = new AktivitasId();
        $jenisPelatihan = $this->getKriteria(self::ID_ASPEK_JENIS_PELATIHAN, $this->formatJenisPelatihan($kueriPelatihan->id_jenis_latih), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$jenisPelatihan];
//        dd($kumpulanElemenPenilaian);

        $pelatihan = Pelatihan::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriPelatihan->nama,
            $kueriPelatihan->id_pelatihan,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '4'
        );

        return $pelatihan;
    }

    public function updateKumpulanPelatihan(array
$kumpulanIdPelatihan, MahasiswaId $idMhs, SkemId $idSkem, string
$tglKlaim,  int $periode)
    {
```

```
        $kumpulanPelatihan = [];
        foreach ($kumpulanIdPelatihan as $idPelatihan) {
            $pelatihan = $this->updatePelatihan($idPelatihan, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanPelatihan, $pelatihan);
        }

        return $kumpulanPelatihan;
    }
}
```

5.2.3.14 MssqlPertukaranMhsRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use App\Modules\Skem\Domain\Exception\PadananBobotNotFoundException;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\PertukaranMhs\PertukaranMhs;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\PertukaranMhsRepository;

use Illuminate\Support\Facades\DB;

class MssqlPertukaranMhsRepository implements PertukaranMhsRepository
{
    const ID_JENIS_PORTOFOLIO = 'pertukaran_mhs';
    const STATUS_VALIDASI = 2;
    const ID_BIDANG_SKEM = 11;

    const ID_ASPEK_PELAKSANAAN = 23;
    const ID_ASPEK_TINGKAT = 24;
```

```
    const ID_ASPEK_DISIPLIN = 25;
    const ID_ASPEK_RENTANG_WAKTU = 26;

    public function requestList(MahasiswaId $idMhs, int $periode)
    {
        $idMhs = $idMhs->id();

        $kueriKumpulanPertukaranMhs = DB::table('dbo.pertukaran_mhs')
            ->where('dbo.pertukaran_mhs.id_mhs', '=', $idMhs)
            ->where('dbo.pertukaran_mhs.status_validasi',                    '=',
self::STATUS_VALIDASI)
            ->whereNotIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query) use ($idMhs) {
                $query->select('dbo.kegiatan_skem.id_portofolio')
                    ->from('dbo.kegiatan_skem')
                    ->join('dbo.skem',        'dbo.kegiatan_skem.id_skem',        '=',
'dbo.skem.id_skem')
                    ->where('dbo.skem.id_mhs', '=', $idMhs)
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',              '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',                   '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode )
            ->select('dbo.pertukaran_mhs.nama',
'dbo.pertukaran_mhs.id_tukar_mhs')
            ->get();
        return $kueriKumpulanPertukaranMhs;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }
```

```
   public function get(string $idPertukaranMhs, MahasiswaId $idMhs, int
$periode)
   {
      $idMhs = $idMhs->id();

      $kueriPertukaranMhs = DB::table('dbo.pertukaran_mhs')
         ->where('dbo.pertukaran_mhs.id_mhs', '=', $idMhs )
         ->where('dbo.pertukaran_mhs.id_tukar_mhs',                    '=',
$idPertukaranMhs )
         ->where('dbo.pertukaran_mhs.status_validasi',                 '=',
self::STATUS_VALIDASI)
         ->whereNotIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem',       'dbo.kegiatan_skem.id_skem',       '=',
'dbo.skem.id_skem')
               ->where('dbo.skem.id_mhs', '=', $idMhs)
               ->where('dbo.kegiatan_skem.id_jenis_portofolio',           '=',
self::ID_JENIS_PORTOFOLIO)
               ->where('dbo.kegiatan_skem.id_bidang_skem',                '=',
self::ID_BIDANG_SKEM);
         })
         ->whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode )
         ->select('dbo.pertukaran_mhs.id_tukar_mhs',
            'dbo.pertukaran_mhs.nama',
            'dbo.pertukaran_mhs.tgl_mulai',
            'dbo.pertukaran_mhs.tgl_selesai',
            'dbo.pertukaran_mhs.tempat',
            'dbo.pertukaran_mhs.skala',
            'dbo.pertukaran_mhs.status_validasi',
            'dbo.pertukaran_mhs.tgl_validasi',
            'dbo.pertukaran_mhs.id_validator',
            'dbo.pertukaran_mhs.nama_validator',
            'dbo.pertukaran_mhs.catatan',
            'dbo.pertukaran_mhs.id_mhs',
            DB::raw('DATEDIFF(MONTH,
dbo.pertukaran_mhs.tgl_mulai,
```

```
            dbo.pertukaran_mhs.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriPertukaranMhs;
  }

  public function formatPelaksanaan()
  {
    return 72; // otomatis individu
  }

  public function formatRentangWaktu(int $rentangWaktu)
  {
    if ($rentangWaktu <= 1 && $rentangWaktu > -1) {
      return 79;
    } elseif ($rentangWaktu <= 3) {
      return 80;
    } elseif ($rentangWaktu <= 6) {
      return 81;
    } elseif ($rentangWaktu <= 12) {
      return 82;
    } elseif ($rentangWaktu > 12) {
      return 83;
    } else {
      throw new PadananBobotNotFoundException('rentang waktu',
'internasionalisasi');
    }
  }

  public function formatBidangIlmu()
  {
    return 78; // otomatis berhubungan
  }

  public function formatSkala(string $skala)
  {
    if ($skala == 'I') {
```

```
            return 76; // internasional
        } elseif ( $skala == 'N') {
            return 75; // nasional
        } else {
            throw new PadananBobotNotFoundException('skala', 'pertukaran
mahasiswa');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
            ->join('ref.aspek_nilai_skem', 'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem', '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
                'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
                'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildPertukaranMhs(string $idPertukaranMhs,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
```

```
        $kueriPertukaranMhs = $this->get($idPertukaranMhs, $idMhs, $periode);
        if ($kueriPertukaranMhs == null) {
            throw new AktivitasIsNullException('pertukaran mhs');
        }
        $idKegiatanSkem = new AktivitasId();

        $pelaksanaan = $this->getKriteria(self::ID_ASPEK_PELAKSANAAN, $this->formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this->formatSkala($kueriPertukaranMhs->skala), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this->formatBidangIlmu(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriPertukaranMhs->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin, $rentangWaktu];


        $pertukaranMhs = PertukaranMhs::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriPertukaranMhs->nama,
            $kueriPertukaranMhs->id_tukar_mhs,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '3'
        );

        return $pertukaranMhs;
    }
```

```
    public function buildKumpulanPertukaranMhs(array
$kumpulanIdPertukaranMhs, MahasiswaId $idMhs, SkemId $idSkem,
string $tglKlaim, int $periode)
    {
        $kumpulanPertukaranMhs = [];
        foreach ($kumpulanIdPertukaranMhs as $idPertukaranMhs) {
            $pertukaranMhs = $this->buildPertukaranMhs($idPertukaranMhs,
$idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanPertukaranMhs, $pertukaranMhs);
        }

        return $kumpulanPertukaranMhs;
    }

    /*
     * UPDATE
     */

    public function requestListForUpdate(MahasiswaId $idMhs, int
$periode, SkemId $idSkem)
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriKumpulanPertukaranMhs = DB::table('dbo.pertukaran_mhs')
            ->where('dbo.pertukaran_mhs.id_mhs', '=', $idMhs)
            ->where('dbo.pertukaran_mhs.status_validasi',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem',
'=', 'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
```

```
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem',         '=',
self::ID_BIDANG_SKEM);
                })
                  -
>whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode );
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query4) use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                        ->from('dbo.kegiatan_skem')
                        ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                        ->where('dbo.skem.id_mhs', '=', $idMhs)
                        ->where('dbo.skem.id_skem', '=', $idSkem)
                        ->where('dbo.kegiatan_skem.id_jenis_portofolio',  '=',
self::ID_JENIS_PORTOFOLIO)
                        ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                                                                          -
>whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.pertukaran_mhs.nama',
'dbo.pertukaran_mhs.id_tukar_mhs')
        ->get();
    return $kueriKumpulanPertukaranMhs;
  }

  public function getForUpdate(string $idPertukaranMhs, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
  {
```

```
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriPertukaranMhs = DB::table('dbo.pertukaran_mhs')
            ->where('dbo.pertukaran_mhs.id_mhs', '=', $idMhs )
            ->where('dbo.pertukaran_mhs.id_tukar_mhs',                    '=',
$idPertukaranMhs )
            ->where('dbo.pertukaran_mhs.status_validasi',                 '=',
self::STATUS_VALIDASI)
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1->whereNotIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        -
>whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode );
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                    $query3->whereIn('dbo.pertukaran_mhs.id_tukar_mhs',
function($query4) use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',  'dbo.kegiatan_skem.id_skem',  '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
```

```
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                });
//                                                                    -
>whereRaw('DATEDIFF(DAY,dbo.pertukaran_mhs.tgl_selesai,
GETDATE()) <= ' . $periode );
            });
        })
        ->select('dbo.pertukaran_mhs.id_tukar_mhs',
          'dbo.pertukaran_mhs.nama',
          'dbo.pertukaran_mhs.tgl_mulai',
          'dbo.pertukaran_mhs.tgl_selesai',
          'dbo.pertukaran_mhs.tempat',
          'dbo.pertukaran_mhs.skala',
          'dbo.pertukaran_mhs.status_validasi',
          'dbo.pertukaran_mhs.tgl_validasi',
          'dbo.pertukaran_mhs.id_validator',
          'dbo.pertukaran_mhs.nama_validator',
          'dbo.pertukaran_mhs.catatan',
          'dbo.pertukaran_mhs.id_mhs',
          DB::raw('DATEDIFF(MONTH,
dbo.pertukaran_mhs.tgl_mulai,
           dbo.pertukaran_mhs.tgl_selesai) as rentang_waktu') )
        ->first();

    return $kueriPertukaranMhs;
  }

  public function updatePertukaranMhs(string $idPertukaranMhs,
MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
  {
    $kueriPertukaranMhs = $this->getForUpdate($idPertukaranMhs,
$idMhs, $periode, $idSkem);
    if ($kueriPertukaranMhs == null) {
      throw new AktivitasIsNullException('pertukaran mhs');
    }
    $idKegiatanSkem = new AktivitasId();
```

```
        $pelaksanaan = $this->getKriteria(self::ID_ASPEK_PELAKSANAAN, $this->formatPelaksanaan(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $tingkat = $this->getKriteria(self::ID_ASPEK_TINGKAT, $this->formatSkala($kueriPertukaranMhs->skala), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $disiplin = $this->getKriteria(self::ID_ASPEK_DISIPLIN, $this->formatBidangIlmu(), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriPertukaranMhs->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $kumpulanElemenPenilaian = [$pelaksanaan, $tingkat, $disiplin, $rentangWaktu];


        $pertukaranMhs  = PertukaranMhs::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriPertukaranMhs->nama,
            $kueriPertukaranMhs->id_tukar_mhs,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            '3' // karena posisi pasti peserta
        );

        return $pertukaranMhs;
    }

    public function updateKumpulanPertukaranMhs(array $kumpulanIdPertukaranMhs, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kumpulanPertukaranMhs = [];
        foreach ($kumpulanIdPertukaranMhs as $idPertukaranMhs) {
```

```
        $pertukaranMhs = $this->updatePertukaranMhs($idPertukaranMhs, $idMhs, $idSkem, $tglKlaim, $periode);
            array_push($kumpulanPertukaranMhs, $pertukaranMhs);
        }

        return $kumpulanPertukaranMhs;
    }

}
```

5.2.3.15 MssqlSemesterRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Repositories\SemesterRepository;

use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Semester\Semester;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;

use Illuminate\Support\Facades\DB;

class MssqlSemesterRepository implements SemesterRepository
{

    public function getSemuaSemester(MahasiswaId $idMhs)
    {
        $idMahasiswa = $idMhs->id();

        $kueriKumpulanSemester = DB::table('ref.semester')
            ->where('ref.semester.id_thn_ajar', '>=', function($query) use ($idMahasiswa) {
                $query->select('thn_angkatan')
                    ->from('dbo.mahasiswa')
```

```
                ->where('dbo.mahasiswa.id_mhs', '=', $idMahasiswa)
                ->first();
        })
//          ->where('ref.semester.tgl_selesai', '<=', DB::raw('GETDATE()'))
        ->whereRaw( 'ref.semester.tgl_mulai <= GETDATE()')
        ->oldest('ref.semester.tgl_mulai')
        ->get();

    $kumpulanSemester = [];
    $semesterKe = 1;
    foreach ($kueriKumpulanSemester as $kueriSemester) {
        $semester = new Semester($idMhs, new
SemesterId($kueriSemester->id_smt), $semesterKe, $kueriSemester-
>nama);
        array_push($kumpulanSemester, $semester);
        $semesterKe += 1;
    }

    return $kumpulanSemester;
  }

  public function getSemester(MahasiswaId $idMhs, SemesterId $idSmt)
  {
    $idMahasiswa = $idMhs->id();
    $idSemester = $idSmt->id();

    $kueriSemester = DB::table('ref.semester')
        ->where('ref.semester.id_smt', '=', $idSemester)
        ->first();

    $kueriSemuaSemester = DB::table('ref.semester')
        ->where('ref.semester.id_thn_ajar', '>=', function($query) use
($idMahasiswa) {
            $query->select('thn_angkatan')
                ->from('dbo.mahasiswa')
                ->where('dbo.mahasiswa.id_mhs', '=', $idMahasiswa)
                ->first();
        })
//          ->where('ref.semester.tgl_selesai', '<=', DB::raw('GETDATE()'))
```

```
        ->where( 'ref.semester.tgl_mulai' ,'<=', function($query) use
($idSemester){
            $query->select('tgl_mulai')
               ->from('ref.semester')
               ->where('ref.semester.id_smt',  '=', $idSemester)
               ->first();
         })
         ->get();
      $semesterKe = count($kueriSemuaSemester);
      $semester = new Semester($idMhs, $idSmt, $semesterKe,
$kueriSemester->nama);

      return $semester;
   }

   public function getSemuaSemesterAnakWali()
   {
      $kueriKumpulanSemester = DB::table('ref.semester')
         ->whereRaw('ref.semester.tgl_mulai <= GETDATE()')
         ->latest('ref.semester.tgl_mulai')
         ->limit(2)
         ->get();

      return $kueriKumpulanSemester;

   }

   public function getSemesterSekarang()
   {
      $kueriSemesterSekarang = DB::table('ref.semester')
         ->whereRaw('ref.semester.tgl_mulai <= GETDATE()')
         ->whereRaw('ref.semester.tgl_selesai >= GETDATE()')
         ->first();
      return $kueriSemesterSekarang;
   }

   public function getSemesterById($idSmt)
```

```
    {
        $kueriSemester = DB::table('ref.semester')
            ->where('ref.semester.id_smt', '=', $idSmt)
            ->first();
        return $kueriSemester;
    }

//    public function getCurrentSemester($idMhs)
//    {
//        $arrSmt = DB::table('ref.semester')
//                ->where('ref.semester.id_thn_ajar', '>=', function($query) use
($idMhs) {
//                $query->select('thn_angkatan')
//                    ->from('dbo.mahasiswa')
//                    ->where('dbo.mahasiswa.id_mhs', '=', $idMhs)
//                    ->first();
//            })
////          ->where('ref.semester.tgl_selesai', '<=', DB::raw('GETDATE()'))
//          ->whereRaw( 'ref.semester.tgl_mulai <= GETDATE()')
//          ->latest('ref.semester.tgl_mulai')
//          ->get();
//
//        $count_smt = count($arrSmt);
//        $curr_smt = $arrSmt[0];
//        $curr_smt->semesterKe = $count_smt;
//
//        return $curr_smt;
//    }

}
```

5.2.3.16 MssqlSkemRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\DuplicateSkemException;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
```

```
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\Mahasiswa;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Validator\ValidatorId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Repositories\SkemRepository;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;

use Illuminate\Support\Facades\DB;

class MssqlSkemRepository implements SkemRepository
{
   public function add(Skem $skem)
   {
      DB::transaction(function () use ($skem) {

      $exist = DB::table('dbo.skem')
         ->where('dbo.skem.id_mhs', $skem->getIdMhs()->id())
         ->where('dbo.skem.id_smt', $skem->getIdSmt()->id())
         ->exists();
      if($exist == true) {
         throw new DuplicateSkemException($skem->getIdMhs()->id(),
$skem->getIdSmt()->id());
      }
         DB::table('dbo.skem')->insert(
            ['id_skem' => $skem->getIdSkem()->id(),
               'id_mhs' => $skem->getIdMhs()->id(),
               'id_smt' => $skem->getIdSmt()->id(),
               'semester_ke' => $skem->getSemesterKe(),
               'status_ajuan' => $skem->getStatusAjuan(),
               'status_lulus' => $skem->getStatusLulus(),
               'total_skem_a' => $skem->getTotalProdukA(),
               'total_skem_b' => $skem->getTotalProdukB(),
               'total_skem_c' => $skem->getTotalProdukC(),
               'total_skem_d' => $skem->getTotalProdukD(),
```

```
                'total_nilai_skem' => $skem->getTotalProdukSkem(),
                'ip_skem' => $skem->getIpSkem(),
                'updater' => $skem->getIdMhs()->id()
            ]
        );
        $kumpulanAktivitas = $skem->getKumpulanAktivitas();
        foreach ($kumpulanAktivitas as $aktivitas) {
            DB::table('dbo.kegiatan_skem')->insert(
                ['id_kegiatan_skem'   =>   $aktivitas->getIdKegiatanSkem()-
>id(),
                    'id_skem' => $skem->getIdSkem()->id(),
                    'id_bidang_skem' => $aktivitas->getIdBidangSkem(),
                    'tgl_klaim' => $aktivitas->getTglKlaim(),
                    'nama_kegiatan' => $aktivitas->getNamaKegiatan(),
                    'kredit_skem' => $aktivitas->getKreditSkem(),
                    'nilai_angka' => $aktivitas->getNilaiAngka(),
                    'nilai_huruf' => $aktivitas->getNilaiHuruf(),
                    'produk_skem' => $aktivitas->getProdukSkem(),
                    'id_portofolio' => $aktivitas->getIdPortofolio(),
                    'id_jenis_portofolio' => $aktivitas->getIdJenisPortofolio(),
                    'updater' => $skem->getIdMhs()->id()
                ]
            );
            if (count($aktivitas->getKumpulanElemenPenilaian()) > 0) {
                $kumpulanElemenPenilaian          =          $aktivitas-
>getKumpulanElemenPenilaian();
                foreach ($kumpulanElemenPenilaian as $elemenPenilaian) {
                    DB::table('dbo.komponen_nilai')->insert(
                        ['id_kegiatan_skem'          =>          $aktivitas-
>getIdKegiatanSkem()->id(),
                            'id_aspek_nilai'        =>        $elemenPenilaian-
>getIdAspekNilai(),
                            'id_kriteria_skem'       =>       $elemenPenilaian-
>getIdKriteriaSkem(),
                            'bobot' => $elemenPenilaian->getBobot(),
                            'updater' => $skem->getIdMhs()->id()
                        ]
                    );
                }
```

```
            }
        }
    });
}

public function update(Skem $skem)
{
    DB::transaction(function () use ($skem) {
        DB::table('dbo.skem')
            ->where('dbo.skem.id_skem', '=', $skem->getIdSkem()->id())
            ->update(['status_ajuan' => $skem->getStatusAjuan(),
                    'status_lulus' => $skem->getStatusLulus(),
                    'total_skem_a' => $skem->getTotalProdukA(),
                    'total_skem_b' => $skem->getTotalProdukB(),
                    'total_skem_c' => $skem->getTotalProdukC(),
                    'total_skem_d' => $skem->getTotalProdukD(),
                    'total_nilai_skem' => $skem->getTotalProdukSkem(),
                    'ip_skem' => $skem->getIpSkem(),
                    'updater' => $skem->getIdMhs()->id()
                ]
            );
        $this->deleteSemuaKegiatanSkem($skem->getIdSkem());
        $kumpulanAktivitas = $skem->getKumpulanAktivitas();

        foreach ($kumpulanAktivitas as $aktivitas) {
            DB::table('dbo.kegiatan_skem')->insert(
                ['id_kegiatan_skem'   =>   $aktivitas->getIdKegiatanSkem()-
>id(),
                    'id_skem' => $skem->getIdSkem()->id(),
                    'id_bidang_skem' => $aktivitas->getIdBidangSkem(),
                    'tgl_klaim' => $aktivitas->getTglKlaim(),
                    'nama_kegiatan' => $aktivitas->getNamaKegiatan(),
                    'kredit_skem' => $aktivitas->getKreditSkem(),
                    'nilai_angka' => $aktivitas->getNilaiAngka(),
                    'nilai_huruf' => $aktivitas->getNilaiHuruf(),
                    'produk_skem' => $aktivitas->getProdukSkem(),
                    'id_portofolio' => $aktivitas->getIdPortofolio(),
```

```
                'id_jenis_portofolio' => $aktivitas->getIdJenisPortofolio(),
                'updater' => $skem->getIdMhs()->id()
            ]
        );
        if (count($aktivitas->getKumpulanElemenPenilaian()) > 0) {
            $kumpulanElemenPenilaian = $aktivitas->getKumpulanElemenPenilaian();
            foreach ($kumpulanElemenPenilaian as $elemenPenilaian) {
                DB::table('dbo.komponen_nilai')->insert(
                    ['id_kegiatan_skem' => $aktivitas->getIdKegiatanSkem()->id(),
                    'id_aspek_nilai' => $elemenPenilaian->getIdAspekNilai(),
                    'id_kriteria_skem' => $elemenPenilaian->getIdKriteriaSkem(),
                    'bobot' => $elemenPenilaian->getBobot(),
                    'updater' => $skem->getIdMhs()->id()
                ]
                );
            }
        }
    });
  }

  public function deleteSemuaKegiatanSkem(SkemId $idSkem)
  {
    $idSkem = $idSkem->id();
    DB::transaction(function () use ($idSkem) {
      $kumpulanKegiatanSkem = DB::table('dbo.kegiatan_skem')
        ->where('dbo.kegiatan_skem.id_skem', '=', $idSkem)
        ->get();

      foreach ($kumpulanKegiatanSkem as $kegiatanSkem) {
        DB::table('dbo.komponen_nilai')
          ->where('dbo.komponen_nilai.id_kegiatan_skem', '=', $kegiatanSkem->id_kegiatan_skem)
          ->delete();
      }
```

```
            DB::table('dbo.kegiatan_skem')
                ->where('dbo.kegiatan_skem.id_skem', '=', $idSkem)
                ->delete();
        });
    }

    public function delete(SkemId $idSkem)
    {
        $idSkem = $idSkem->id();

        DB::transaction(function () use ($idSkem) {
            $kumpulanKegiatanSkem = DB::table('dbo.kegiatan_skem')
                ->where('dbo.kegiatan_skem.id_skem', '=', $idSkem)
                ->get();

            foreach ($kumpulanKegiatanSkem as $kegiatanSkem) {
                DB::table('dbo.komponen_nilai')
                    ->where('dbo.komponen_nilai.id_kegiatan_skem', '=',
$kegiatanSkem->id_kegiatan_skem)
                    ->delete();
            }
            DB::table('dbo.kegiatan_skem')
                ->where('dbo.kegiatan_skem.id_skem', '=', $idSkem)
                ->delete();
            DB::table('dbo.skem')->where('dbo.skem.id_skem', '=', $idSkem)
                ->delete();
        });
    }

    public function getKegiatanSkemView(AktivitasId $idKegiatanSkem)
    {
        $idKegiatanSkem = $idKegiatanSkem->id();
        $kueriKegiatanSkem = DB::table('dbo.kegiatan_skem')
            ->where('dbo.kegiatan_skem.id_kegiatan_skem', '=',
$idKegiatanSkem)
            ->first();
```

```
        $kegiatanSkem = Aktivitas::makeView(
            new AktivitasId($kueriKegiatanSkem->id_kegiatan_skem),
            new SkemId($kueriKegiatanSkem->id_skem),
            $kueriKegiatanSkem->id_bidang_skem,
            $kueriKegiatanSkem->tgl_klaim,
            $kueriKegiatanSkem->nama_kegiatan,
            $kueriKegiatanSkem->id_portofolio,
            $kueriKegiatanSkem->id_jenis_portofolio,
            (int) $kueriKegiatanSkem->kredit_skem,
            (int) $kueriKegiatanSkem->nilai_angka,
            $kueriKegiatanSkem->nilai_huruf,
            (int) $kueriKegiatanSkem->produk_skem
        );

        return $kegiatanSkem;
    }

    public function getKegiatanSkemDenganKumpulanElemenPenilaian(AktivitasId $idKegiatanSkem)
    {
        $idKegiatanSkemOriginal = $idKegiatanSkem;

        $idKegiatanSkem = $idKegiatanSkem->id();
        $kueriKegiatanSkem = DB::table('dbo.kegiatan_skem')
            ->join('dbo.skem', 'dbo.skem.id_skem', '=', 'dbo.kegiatan_skem.id_skem')
            ->where('dbo.kegiatan_skem.id_kegiatan_skem', '=', $idKegiatanSkem)
            ->first();

        $kueriKumpulanElemenPenilaian = DB::table('dbo.komponen_nilai')
            ->join('ref.aspek_nilai_skem', 'dbo.komponen_nilai.id_aspek_nilai', 'ref.aspek_nilai_skem.id_aspek_nilai')
            ->join('ref.kriteria_skem', 'dbo.komponen_nilai.id_kriteria_skem', 'ref.kriteria_skem.id_kriteria_skem')
```

```
        ->where('dbo.komponen_nilai.id_kegiatan_skem', '=',
$idKegiatanSkem)
        ->select('dbo.komponen_nilai.*', 'ref.aspek_nilai_skem.nama as
aspek_nilai', 'ref.kriteria_skem.nama as kriteria_skem')
        ->get();

    $kumpulanElemenPenilaian =[];
    foreach($kueriKumpulanElemenPenilaian as
$kueriElemenPenilaian) {
        $elemenPenilaian = ElemenPenilaian::makeView(
            $idKegiatanSkemOriginal,
            $kueriElemenPenilaian->id_kriteria_skem,
            $kueriElemenPenilaian->id_aspek_nilai,
            $kueriElemenPenilaian->aspek_nilai,
            $kueriElemenPenilaian->kriteria_skem,
            $kueriElemenPenilaian->bobot
        );
        array_push($kumpulanElemenPenilaian, $elemenPenilaian);
    }

    $kegiatanSkem = Aktivitas::makeView(
        new AktivitasId($kueriKegiatanSkem->id_kegiatan_skem),
        new SkemId($kueriKegiatanSkem->id_skem),
        $kueriKegiatanSkem->id_bidang_skem,
        $kueriKegiatanSkem->tgl_klaim,
        $kueriKegiatanSkem->nama_kegiatan,
        $kueriKegiatanSkem->id_portofolio,
        $kueriKegiatanSkem->id_jenis_portofolio,
        $kueriKegiatanSkem->nilai_huruf,
        (int) $kueriKegiatanSkem->nilai_angka,
        (int) $kueriKegiatanSkem->kredit_skem,
        (int) $kueriKegiatanSkem->produk_skem
    );
    $kegiatanSkem-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);

    $kegiatanSkem->idMahasiswa = $kueriKegiatanSkem->id_mhs;
```

```
        $kegiatanSkem->idSmt = $kueriKegiatanSkem->id_smt;

        return $kegiatanSkem;
    }

    public function getSemuaKegiatanSkem(skemId $idSkem)
    {
        $idSkemOriginal = $idSkem;
        $idSkem = $idSkem->id();

        $kueriKumpulanKegiatanSkem = DB::table('dbo.kegiatan_skem')
            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
            ->where('dbo.skem.id_skem', '=', $idSkem)
            ->get();

        $kumpulanKegiatanSkem = [];
        foreach ($kueriKumpulanKegiatanSkem as $kueriKegiatanSkem)
        {
            $kegiatanSkem = Aktivitas::makeView(
                new AktivitasId($kueriKegiatanSkem->id_kegiatan_skem),
                $idSkemOriginal,
                $kueriKegiatanSkem->id_bidang_skem,
                $kueriKegiatanSkem->tgl_klaim,
                $kueriKegiatanSkem->nama_kegiatan,
                $kueriKegiatanSkem->id_portofolio,
                $kueriKegiatanSkem->id_jenis_portofolio,
                $kueriKegiatanSkem->nilai_huruf,
                (int)$kueriKegiatanSkem->nilai_angka,
                (int)$kueriKegiatanSkem->kredit_skem,
                (int)$kueriKegiatanSkem->produk_skem
            );
            array_push($kumpulanKegiatanSkem, $kegiatanSkem);
        }

        return $kumpulanKegiatanSkem;
    }
```

```
    public function getSemuaIdPortofolioKegiatanSkemForEdit(SkemId
$idSkem)
    {
        $kumpulanKegiatanSkem = $this-
>getSemuaKegiatanSkem($idSkem);
        $idSkem = $idSkem->id();
        $kumpulanIdPortofolioKegiatanSkem =
[[],[],[],[],[],[],[],[],[],[],[],[],[],[]];
        foreach ($kumpulanKegiatanSkem as $kegiatanSkem) {
            $kumpulanIdPortofolioKegiatanSkem[$kegiatanSkem-
>getIdBidangSkem()][] = $kegiatanSkem->getIdPortofolio();
        }
        return $kumpulanIdPortofolioKegiatanSkem;
    }

    public function getSkem(SkemId $idSkem)
    {
        $idSkem = $idSkem->id();

        $kueriSkem = DB::table('dbo.skem')
            ->where('dbo.skem.id_skem', '=', $idSkem)
            ->first();

        $skem = Skem::makeView(
            new SkemId($kueriSkem->id_skem),
            new MahasiswaId($kueriSkem->id_mhs),
            new SemesterId($kueriSkem->id_smt),
            $kueriSkem->semester_ke,
            (int)$kueriSkem->status_ajuan,
            $kueriSkem->status_lulus,
            (int)$kueriSkem->total_skem_a,
            (int)$kueriSkem->total_skem_b,
            (int)$kueriSkem->total_skem_c,
            (int)$kueriSkem->total_skem_d,
            (int)$kueriSkem->total_nilai_skem,
            (float)$kueriSkem->ip_skem,
            $kueriSkem->komentar
```

```
        );
        return $skem;
    }

    public function getSkemDenganKumpulanKegiatanSkem(skemId
$idSkem)
    {
        $kumpulanKegiatanSkem = $this-
>getSemuaKegiatanSkem($idSkem);

        $idSkem = $idSkem->id();
        $kueriSkem = DB::table('dbo.skem')
            ->where('dbo.skem.id_skem', '=', $idSkem)
            ->first();

        $skem = Skem::makeView(
            new SkemId($kueriSkem->id_skem),
            new MahasiswaId($kueriSkem->id_mhs),
            new SemesterId($kueriSkem->id_smt),
            (int)$kueriSkem->semester_ke,
            (int)$kueriSkem->status_ajuan,
            $kueriSkem->status_lulus,
            (int)$kueriSkem->total_skem_a,
            (int)$kueriSkem->total_skem_b,
            (int)$kueriSkem->total_skem_c,
            (int)$kueriSkem->total_skem_d,
            (int)$kueriSkem->total_nilai_skem,
            (float)$kueriSkem->ip_skem,
            $kueriSkem->komentar
        );
        $skem->setKumpulanAktivitas($kumpulanKegiatanSkem);

//        dd($skem);
        return $skem;
    }

    public function getSemuaSkemView(MahasiswaId  $idMhs)
    {
        $idMahasiswa = $idMhs->id();
```

```
        $kueriKumpulanSkem = DB::table('dbo.skem')
            ->where('dbo.skem.id_mhs', '=', $idMahasiswa)
            ->oldest('dbo.skem.semester_ke')
            ->get();

        $kumpulanSkem = [];
        foreach ($kueriKumpulanSkem as $kueriSkem) {
            $skem = Skem::makeView(
                new SkemId($kueriSkem->id_skem),
                $idMhs,
                new SemesterId($kueriSkem->id_smt),
                $kueriSkem->semester_ke,
                $kueriSkem->status_ajuan,
                $kueriSkem->status_lulus,
                $kueriSkem->total_skem_a,
                $kueriSkem->total_skem_b,
                $kueriSkem->total_skem_c,
                $kueriSkem->total_skem_d,
                $kueriSkem->total_nilai_skem,
                $kueriSkem->ip_skem,
                $kueriSkem->komentar
            );
            array_push($kumpulanSkem, $skem);
        }

        return $kumpulanSkem;
    }

    public                                                    function
getSemuaSkemDenganKumpulanKegiatanSkem(MahasiswaId $idMhs)
    {
        $kumpulanIdSkem = $this->getSemuaIdSkem($idMhs);
        $kumpulanSkem = [];
        foreach ($kumpulanIdSkem as $idSkem) {
            $skem    =    $this->getSkemDenganKumpulanKegiatanSkem(new
SkemId($idSkem));
```

```
            array_push($kumpulanSkem, $skem);
        }
        return $kumpulanSkem;
    }

    public function getSemuaIdSkem(MahasiswaId $idMhs)
    {
        $idMhs = $idMhs->id();
        $kueriKumpulanSkem = DB::table('dbo.skem')
            ->where('dbo.skem.id_mhs', '=', $idMhs)
            ->select('dbo.skem.id_skem')
            ->get();

        $kumpulanIdSkem = [];
        foreach($kueriKumpulanSkem as $kueriSkem) {
            array_push($kumpulanIdSkem, $kueriSkem->id_skem);
        }

        return $kumpulanIdSkem;
    }

    public function updateStatus(skemId $idSkem, int $statusAjuan,
mahasiswaId $idUpdater)
    {
        $idSkem = $idSkem->id();
        $idUpdater = $idUpdater->id();
        $affected = DB::table('dbo.skem')
            ->where('dbo.skem.id_skem', '=', $idSkem)
            ->update(['status_ajuan' => $statusAjuan, 'updater' =>
$idUpdater]);

//        dd($affected);
    }

    /*
     * VALIDATOR
     */
```

```
   public function getSkemAnakWali(MahasiswaId $idMhs, SemesterId
$idSmt)
   {
      $kueriSkem = DB::table('dbo.skem')
         ->where('dbo.skem.id_smt', $idSmt->id())
         ->where('dbo.skem.id_mhs', $idMhs->id())
         ->first();

      if ($kueriSkem != null) {
         $skem = Skem::makeView(
            new SkemId($kueriSkem->id_skem),
            new MahasiswaId($kueriSkem->id_mhs),
            new SemesterId($kueriSkem->id_smt),
            $kueriSkem->semester_ke,
            (int)$kueriSkem->status_ajuan,
            $kueriSkem->status_lulus,
            (int)$kueriSkem->total_skem_a,
            (int)$kueriSkem->total_skem_b,
            (int)$kueriSkem->total_skem_c,
            (int)$kueriSkem->total_skem_d,
            (int)$kueriSkem->total_nilai_skem,
            (float)$kueriSkem->ip_skem,
            $kueriSkem->komentar
         );
      } else {
         $skem = null;
      }

      return $skem;
   }

   public                                                    function
getSkemAnakWaliDenganKumpulanKegiatan(MahasiswaId      $idMhs,
SemesterId $idSmt)
   {
      $skem = $this->getSkemAnakWali($idMhs, $idSmt);
      if ($skem != null) {
```

```
        $kumpulanKegiatanSkem = $this->getSemuaKegiatanSkem($skem->getIdSkem());
        $skem->setKumpulanAktivitas($kumpulanKegiatanSkem);
      }

      return $skem;
   }

   public function updateStatusAndMessage($idSkem, $statusAjuan, $idUpdater, $komentar)
   {
      $affected = DB::table('dbo.skem')
         ->where('dbo.skem.id_skem', '=', $idSkem)
         ->update([
            'status_ajuan' => $statusAjuan,
            'updater' => $idUpdater,
            'komentar' => $komentar,
            'tgl_penilaian' => date('Y-m-d'),
            'id_penilai' => $idUpdater
            ]);
      return $affected;
   }

}
```

5.2.3.17 MssqlWirausahaRepository

```
<?php

namespace App\Modules\Skem\Infrastructure\Persistence;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use App\Modules\Skem\Domain\Exception\ObjectNotFoundException;
use App\Modules\Skem\Domain\Exception\PadananBobotNotFoundException;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
```

```
use App\Modules\Skem\Domain\Model\Skem\SkemId;
use App\Modules\Skem\Domain\Model\Wirausaha\Wirausaha;
use App\Modules\Skem\Domain\Repositories\WirausahaRepository;

use Illuminate\Support\Facades\DB;

class MssqlWirausahaRepository implements WirausahaRepository
{
   const ID_JENIS_PORTOFOLIO = 'wirausaha';
   const STATUS_VALIDASI = 2;
   const ID_BIDANG_SKEM = 3;

   const ID_ASPEK_JUMLAH_PELAKU = 7;
   const ID_ASPEK_BADAN_HUKUM = 8;
   const ID_ASPEK_RENTANG_WAKTU = 9;
   const ID_ASPEK_BIDANG_ILMU = 10;

   public function requestList(MahasiswaId $idMhs, int $periode)
   {
      $idMhs = $idMhs->id();

      $kueriKumpulanWirausaha = DB::table('dbo.wirausaha')
         ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=',
'dbo.pelaku_usaha.id_wirausaha')
         ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
         ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs )
         ->where('dbo.wirausaha.status_validasi', '=',
self::STATUS_VALIDASI)
         ->where('dbo.lap_keuangan_usaha.status_validasi', '=',
self::STATUS_VALIDASI )
         ->whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
               ->from('dbo.kegiatan_skem')
               ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
```

```
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
            })
            ->whereRaw('DATEPART(year,GETDATE()) -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365)
//            ->where(function ($query) use ($periode) {
//                $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                    ->orWhere(function($query) use ($periode) {
//                    $query->whereNotNull('dbo.wirausaha.tgl_tutup')
//                      ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//                    });
//            })
            ->select('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
'dbo.wirausaha.id_wirausaha', 'dbo.lap_keuangan_usaha.thn_laporan' ,
'dbo.wirausaha.nama')
            ->get();

        return $kueriKumpulanWirausaha;
    }

    public function countRequestList(MahasiswaId $idMhs, int $periode)
    {
        $requestList = $this->requestList($idMhs, $periode);
        return count($requestList);
    }

    public function get(string $idLapkeuUsaha, MahasiswaId $idMhs, int
$periode)
    {
        $idMhs = $idMhs->id();

        $kueriWirausaha = DB::table('dbo.wirausaha')
            ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=',
'dbo.pelaku_usaha.id_wirausaha')
```

```
        ->join('ref.jenis_badan_hukum', 'dbo.wirausaha.id_jenis_bh', '=',
'ref.jenis_badan_hukum.id_jenis_bh')
        ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
        ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs)
        ->where('dbo.lap_keuangan_usaha.id_lapkeu_usaha', '=',
$idLapkeuUsaha)
        ->where('dbo.wirausaha.status_validasi', '=',
self::STATUS_VALIDASI)
        ->where('dbo.lap_keuangan_usaha.status_validasi', '=',
self::STATUS_VALIDASI )
        ->whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query) use ($idMhs) {
            $query->select('dbo.kegiatan_skem.id_portofolio')
                ->from('dbo.kegiatan_skem')
                ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                ->where('dbo.skem.id_mhs', '=', $idMhs)
                ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
        })
        ->whereRaw('DATEPART(year,GETDATE()) -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365)
//          ->where(function ($query) use($periode) {
//              $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                  ->orWhere(function($query) use($periode) {
//                      $query->whereNotNull('dbo.wirausaha.tgl_tutup')
//                      ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//                  });
//          })
        ->latest('dbo.lap_keuangan_usaha.thn_laporan')
        ->select('dbo.wirausaha.id_wirausaha',
            'dbo.wirausaha.nama',
            'dbo.wirausaha.tgl_berdiri',
```

```
            'dbo.wirausaha.tempat_kedudukan',
            'dbo.wirausaha.bidang_usaha',
            'dbo.wirausaha.jml_modal_usaha',
            'dbo.wirausaha.tgl_tutup',
            'dbo.wirausaha.is_regu',
            'dbo.wirausaha.jml_pelaku',
            'dbo.wirausaha.is_bidang_ilmu_berhubungan',
            'dbo.wirausaha.id_jenis_bh',
            'dbo.wirausaha.id_prog_wirausaha',
            'dbo.wirausaha.status_validasi',
            'dbo.wirausaha.tgl_validasi',
            'dbo.wirausaha.id_validator',
            'dbo.wirausaha.nama_validator',
            'dbo.wirausaha.catatan',
            'dbo.pelaku_usaha.id_mhs as id_mhs',
            'dbo.pelaku_usaha.is_owner as owner',
            'dbo.lap_keuangan_usaha.*',
            'ref.jenis_badan_hukum.nama as jenis_badan_hukum',
            DB::raw('DATEDIFF(MONTH,       dbo.wirausaha.tgl_berdiri,
ISNULL(dbo.wirausaha.tgl_tutup, GETDATE())) as rentang_waktu'))
        ->first();

      return $kueriWirausaha;
   }

   public function formatJumlahPelaku(int $jumlahPelaku)
   {
      if($jumlahPelaku == 1 ) {
         return 25;
      } elseif ($jumlahPelaku > 1) {
         return 26;
      } else {
         throw new PadananBobotNotFoundException('jumlah pelaku',
'wirausaha');
      }
   }

   public function formatRentangWaktu(int $rentangWaktu)
   {
```

```
        if ($rentangWaktu <= 12 && $rentangWaktu > -1) {
            return 30;
        } elseif ($rentangWaktu > 12) {
            return 31;
        } else {
            throw new PadananBobotNotFoundException('rentang waktu',
'wirausaha');
        }
    }

    public function formatBidangIlmu(int $bidangIlmu)
    {
        if ($bidangIlmu > 0) {
            return 33;
        } else {
            return 32;
        }
    }

    public function formatBadanHukum(int $idBadanHukum)
    {
        if ($idBadanHukum == 1) {
            return 27;
        } elseif ($idBadanHukum == 2 || $idBadanHukum == 3) {
            return 28;
        } elseif ($idBadanHukum == 4) {
            return 29;
        } else {
            throw new PadananBobotNotFoundException('badan hukum',
'wirausaha');
        }
    }

    public function getKriteria(int $idAspek, int $idKriteria, int
$idBidangSkem, AktivitasId $idKegiatanSkem)
    {
        $kueriKriteria = DB::table('ref.kriteria_skem')
```

```
            ->join('ref.aspek_nilai_skem',    'ref.kriteria_skem.id_aspek_nilai',
'=', 'ref.aspek_nilai_skem.id_aspek_nilai' )
            ->where('ref.aspek_nilai_skem.id_aspek_nilai', '=', $idAspek)
            ->where('ref.kriteria_skem.id_kriteria_skem', '=', $idKriteria)
            ->where('ref.aspek_nilai_skem.id_bidang_skem',                '=',
$idBidangSkem)
            ->select('ref.aspek_nilai_skem.nama as aspek_nilai',
               'ref.aspek_nilai_skem.id_bidang_skem as id_bidang_skem',
               'ref.kriteria_skem.*'
            )
            ->first();

        $kriteria = ElemenPenilaian::make(
            $kueriKriteria->id_kriteria_skem,
            $kueriKriteria->id_aspek_nilai,
            $kueriKriteria->aspek_nilai,
            $kueriKriteria->nama,
            $kueriKriteria->bobot
        );

        return $kriteria;
    }

    public function buildWirausaha(string $idLapkeuUsaha, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        $kueriWirausaha = $this->get($idLapkeuUsaha, $idMhs, $periode);
        if ($kueriWirausaha == null) {
            throw new AktivitasIsNullException('wirausaha');
        }
        $idKegiatanSkem = new AktivitasId();

        $jumlahPelaku                =                $this-
>getKriteria(self::ID_ASPEK_JUMLAH_PELAKU,                $this-
>formatJumlahPelaku($kueriWirausaha->jml_pelaku),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $badan_hukum                 =                $this-
>getKriteria(self::ID_ASPEK_BADAN_HUKUM,                $this-
```

```
>formatBadanHukum($kueriWirausaha->id_jenis_bh),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $rentangWaktu = $this-
>getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this-
>formatRentangWaktu($kueriWirausaha->rentang_waktu),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $bidangIlmu = $this-
>getKriteria(self::ID_ASPEK_BIDANG_ILMU, $this-
>formatBidangIlmu($kueriWirausaha->is_bidang_ilmu_berhubungan),
self::ID_BIDANG_SKEM, $idKegiatanSkem);

      $kumpulanElemenPenilaian = [$jumlahPelaku, $badan_hukum,
$rentangWaktu, $bidangIlmu];
      $wirausaha = Wirausaha::make(
         $idKegiatanSkem,
         self::ID_BIDANG_SKEM,
         $tglKlaim,
         $kueriWirausaha->nama,
         $kueriWirausaha->id_lapkeu_usaha,
         self::ID_JENIS_PORTOFOLIO,
         $kumpulanElemenPenilaian,
         $kueriWirausaha->jml_pendapatan
      );

      return $wirausaha;
   }

   public function buildKumpulanWirausaha(array
$kumpulanIdLapkeuUsaha, MahasiswaId $idMhs, SkemId $idSkem,
string $tglKlaim, int $periode)
   {
      if ($kumpulanIdLapkeuUsaha == null ||
count($kumpulanIdLapkeuUsaha) < 1) {
         throw new ObjectNotFoundException("wirausaha");
      }
      $kumpulanWirausaha = [];
      foreach ($kumpulanIdLapkeuUsaha as $idLapkeuUsaha) {
```

```
        $wirausaha = $this->buildWirausaha($idLapkeuUsaha, $idMhs, $idSkem, $tglKlaim, $periode);
        array_push($kumpulanWirausaha, $wirausaha);
      }

      return $kumpulanWirausaha;
   }

   /*
    * UPDATE
    */

   public function requestListForUpdate(MahasiswaId $idMhs, int $periode, SkemId $idSkem)
   {
      $idMhs = $idMhs->id();
      $idSkem = $idSkem->id();

      $kueriKumpulanWirausaha = DB::table('dbo.wirausaha')
         ->join('dbo.pelaku_usaha', 'dbo.wirausaha.id_wirausaha', '=', 'dbo.pelaku_usaha.id_wirausaha')
         ->join('dbo.lap_keuangan_usaha', 'dbo.wirausaha.id_wirausaha', '=', 'dbo.lap_keuangan_usaha.id_wirausaha')
         ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs )
         ->where('dbo.wirausaha.status_validasi', '=', self::STATUS_VALIDASI)
         ->where('dbo.lap_keuangan_usaha.status_validasi', '=', self::STATUS_VALIDASI )
         ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
            $query0->where(function($query1) use ($idMhs, $periode) {
               $query1-
>whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha', function($query2) use ($idMhs) {
                  $query2->select('dbo.kegiatan_skem.id_portofolio')
                     ->from('dbo.kegiatan_skem')
                     ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=', 'dbo.skem.id_skem')
                     ->where('dbo.skem.id_mhs', '=', $idMhs)
```

```
                    ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                    ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                })
                  ->whereRaw('DATEPART(year,GETDATE()) -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365);
            })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
                  $query3-
>whereIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha', function($query4)
use ($idMhs, $idSkem) {
                    $query4->select('dbo.kegiatan_skem.id_portofolio')
                      ->from('dbo.kegiatan_skem')
                      ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                      ->where('dbo.skem.id_mhs', '=', $idMhs)
                      ->where('dbo.skem.id_skem', '=', $idSkem)
                      ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                      ->where('dbo.kegiatan_skem.id_bidang_skem', '=',
self::ID_BIDANG_SKEM);
                    });
//                                                                  -
>whereRaw('DATEDIFF(DAY,dbo.lap_keuangan_usaha.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
        ->select('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
'dbo.wirausaha.id_wirausaha', 'dbo.lap_keuangan_usaha.thn_laporan' ,
'dbo.wirausaha.nama')
        ->get();
      return $kueriKumpulanWirausaha;
  }

  public function getForUpdate(string $idLapkeuUsaha, MahasiswaId
$idMhs, int $periode, SkemId $idSkem)
```

```
    {
        $idMhs = $idMhs->id();
        $idSkem = $idSkem->id();

        $kueriWirausaha = DB::table('dbo.wirausaha')
            ->join('dbo.pelaku_usaha',     'dbo.wirausaha.id_wirausaha',     '=',
'dbo.pelaku_usaha.id_wirausaha')
            ->join('ref.jenis_badan_hukum',  'dbo.wirausaha.id_jenis_bh',  '=',
'ref.jenis_badan_hukum.id_jenis_bh')
            ->join('dbo.lap_keuangan_usaha',    'dbo.wirausaha.id_wirausaha',
'=', 'dbo.lap_keuangan_usaha.id_wirausaha')
            ->where('dbo.pelaku_usaha.id_mhs', '=', $idMhs)
            ->where('dbo.lap_keuangan_usaha.id_lapkeu_usaha',              '=',
$idLapkeuUsaha)
            ->where('dbo.wirausaha.status_validasi',                       '=',
self::STATUS_VALIDASI)
            ->where('dbo.lap_keuangan_usaha.status_validasi',              '=',
self::STATUS_VALIDASI )
            ->where(function ($query0) use ($idMhs, $periode, $idSkem) {
                $query0->where(function($query1) use ($idMhs, $periode) {
                    $query1-
>whereNotIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha',
function($query2) use ($idMhs) {
                        $query2->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem',    'dbo.kegiatan_skem.id_skem',    '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio',     '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',          '=',
self::ID_BIDANG_SKEM);
                    })
                        ->whereRaw('DATEPART(year,GETDATE())                -
dbo.lap_keuangan_usaha.thn_laporan  <= ' . $periode/365);
                })
                ->orWhere(function($query3) use ($idMhs, $idSkem) {
```

```
                    $query3-
>whereIn('dbo.lap_keuangan_usaha.id_lapkeu_usaha', function($query4)
use ($idMhs, $idSkem) {
                        $query4->select('dbo.kegiatan_skem.id_portofolio')
                            ->from('dbo.kegiatan_skem')
                            ->join('dbo.skem', 'dbo.kegiatan_skem.id_skem', '=',
'dbo.skem.id_skem')
                            ->where('dbo.skem.id_mhs', '=', $idMhs)
                            ->where('dbo.skem.id_skem', '=', $idSkem)
                            ->where('dbo.kegiatan_skem.id_jenis_portofolio', '=',
self::ID_JENIS_PORTOFOLIO)
                            ->where('dbo.kegiatan_skem.id_bidang_skem',     '=',
self::ID_BIDANG_SKEM);
                    });
//                                                                          -
>whereRaw('DATEDIFF(DAY,dbo.lap_keuangan_usaha.tgl_selesai,
GETDATE()) <= ' . $periode );
                });
        })
//          ->where(function ($query) use($periode) {
//              $query->WhereNull('dbo.wirausaha.tgl_tutup')
//                  ->orWhere(function($query) use($periode) {
//                      $query->whereNotNull('dbo.wirausaha.tgl_tutup')
//                      ->whereRaw('DATEDIFF(DAY,dbo.wirausaha.tgl_tutup,
GETDATE()) < ' . $periode);
//                  });
//          })
        ->latest('dbo.lap_keuangan_usaha.thn_laporan')
        ->select('dbo.wirausaha.id_wirausaha',
            'dbo.wirausaha.nama',
            'dbo.wirausaha.tgl_berdiri',
            'dbo.wirausaha.tempat_kedudukan',
            'dbo.wirausaha.bidang_usaha',
            'dbo.wirausaha.jml_modal_usaha',
            'dbo.wirausaha.jml_karyawan',
            'dbo.wirausaha.tgl_tutup',
            'dbo.wirausaha.is_regu',
```

```
            'dbo.wirausaha.jml_pelaku',
            'dbo.wirausaha.is_bidang_ilmu_berhubungan',
            'dbo.wirausaha.id_jenis_bh',
            'dbo.wirausaha.id_prog_wirausaha',
            'dbo.wirausaha.status_validasi',
            'dbo.wirausaha.tgl_validasi',
            'dbo.wirausaha.id_validator',
            'dbo.wirausaha.nama_validator',
            'dbo.wirausaha.catatan',
            'dbo.pelaku_usaha.id_mhs as id_mhs',
            'dbo.pelaku_usaha.is_owner as owner',
            'dbo.lap_keuangan_usaha.*',
            'ref.jenis_badan_hukum.nama as jenis_badan_hukum',
            DB::raw('DATEDIFF(MONTH,       dbo.wirausaha.tgl_berdiri,
ISNULL(dbo.wirausaha.tgl_tutup, GETDATE())) as rentang_waktu'))
        ->first();

      return $kueriWirausaha;
   }

   public function updateWirausaha(string $idLapkeuUsaha, MahasiswaId
$idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
   {
      $kueriWirausaha = $this->getForUpdate($idLapkeuUsaha, $idMhs,
$periode, $idSkem);
      if ($kueriWirausaha == null) {
         throw new AktivitasIsNullException('wirausaha');
      }
      $idKegiatanSkem = new AktivitasId();

      $jumlahPelaku                    =                    $this-
>getKriteria(self::ID_ASPEK_JUMLAH_PELAKU,                    $this-
>formatJumlahPelaku($kueriWirausaha->jml_pelaku),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
      $badan_hukum                     =                    $this-
>getKriteria(self::ID_ASPEK_BADAN_HUKUM,                      $this-
>formatBadanHukum($kueriWirausaha->id_jenis_bh),
self::ID_BIDANG_SKEM, $idKegiatanSkem);
```

```
        $rentangWaktu = $this->getKriteria(self::ID_ASPEK_RENTANG_WAKTU, $this->formatRentangWaktu($kueriWirausaha->rentang_waktu), self::ID_BIDANG_SKEM, $idKegiatanSkem);
        $bidangIlmu = $this->getKriteria(self::ID_ASPEK_BIDANG_ILMU, $this->formatBidangIlmu($kueriWirausaha->is_bidang_ilmu_berhubungan), self::ID_BIDANG_SKEM, $idKegiatanSkem);

        $kumpulanElemenPenilaian = [$jumlahPelaku, $badan_hukum, $rentangWaktu, $bidangIlmu];
        $wirausaha = Wirausaha::make(
            $idKegiatanSkem,
            self::ID_BIDANG_SKEM,
            $tglKlaim,
            $kueriWirausaha->nama,
            $kueriWirausaha->id_lapkeu_usaha,
            self::ID_JENIS_PORTOFOLIO,
            $kumpulanElemenPenilaian,
            $kueriWirausaha->jml_pendapatan
        );

        return $wirausaha;
    }

    public function updateKumpulanWirausaha(array $kumpulanIdLapkeuUsaha, MahasiswaId $idMhs, SkemId $idSkem, string $tglKlaim, int $periode)
    {
        if ($kumpulanIdLapkeuUsaha == null || count($kumpulanIdLapkeuUsaha) < 1) {
            throw new ObjectNotFoundException("wirausaha");
        }
        $kumpulanWirausaha = [];
        foreach ($kumpulanIdLapkeuUsaha as $idLapkeuUsaha) {
            $wirausaha = $this->updateWirausaha($idLapkeuUsaha, $idMhs, $idSkem, $tglKlaim, $periode);
```

```
            array_push($kumpulanWirausaha, $wirausaha);
        }

        return $kumpulanWirausaha;
    }

}
```

5.2.4 Domain Model

5.2.4.1 Aktivitas

```
<?php

namespace App\Modules\Skem\Domain\Model\Base;

use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Aktivitas
{
    protected $idKegiatanSkem;
    protected $idSkem;
    protected $idBidangSkem;
    protected $tglKlaim;
    protected $namaKegiatan;
    protected $idPortofolio;
    protected $idJenisPortofolio;

    protected $kreditSkem;
    protected $nilaiAngka;
    protected $nilaiHuruf;
    protected $produkSkem;

    protected $kumpulanElemenPenilaian;
    protected $faktorPengali;
    protected $isView;

    public function __construct(
        AktivitasId $idKegiatanSkem,
```

```
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio
    )
    {
        $this->idKegiatanSkem = $idKegiatanSkem;
        $this->idBidangSkem = $idBidangSkem;
        $this->tglKlaim = $tglKlaim;
        $this->namaKegiatan = $namaKegiatan;
        $this->idPortofolio = $idPortofolio;
        $this->idJenisPortofolio = $idJenisPortofolio;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Aktivitas(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
```

```
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function isView($isView)
    {
        $this->isView = $isView;
    }

    public function setIdSkem(SkemId $idSkem)
    {
        $this->idSkem = $idSkem;
    }

    public function setKumpulanElemenPenilaian(array
$kumpulanElemenPenilaian)
    {
        foreach ($kumpulanElemenPenilaian as $elemenPenilaian) {
            $elemenPenilaian->setIdKegiatanSkem($this->idKegiatanSkem);
        }
        $this->kumpulanElemenPenilaian = $kumpulanElemenPenilaian;
    }

    public function setFaktorPengali($faktorPengali)
    {
        $this->faktorPengali = $faktorPengali;
    }

    public function setKreditSkem(int $kreditSkem)
    {
        $this->kreditSkem = $kreditSkem;
    }
```

```
public function setProdukSkem(int $produkSkem)
{
   $this->produkSkem = $produkSkem;
}

public function setNilaiHuruf(string $nilaiHuruf)
{
   $this->nilaiHuruf = $nilaiHuruf;
}

public function setNilaiAngka(float $nilaiAngka)
{
   $this->nilaiAngka = $nilaiAngka;
}

public function getIdKegiatanSkem()
{
   return $this->idKegiatanSkem;
}

public function getIdSkem()
{
   return $this->idSkem;
}

public function getIdBidangSkem()
{
   return $this->idBidangSkem;
}

public function getTglKlaim()
{
   return $this->tglKlaim;
}

public function getNamaKegiatan()
```

```
    {
        return $this->namaKegiatan;
    }

    public function getIdPortofolio()
    {
        return $this->idPortofolio;
    }

    public function getIdJenisPortofolio()
    {
        return $this->idJenisPortofolio;
    }

    public function getKreditSkem()
    {
        return $this->kreditSkem;
    }

    public function getProdukSkem()
    {
        return $this->produkSkem;
    }

    public function getNilaiHuruf()
    {
        return $this->nilaiHuruf;
    }

    public function getNilaiAngka()

    {
        return $this->nilaiAngka;
    }

    public function getKumpulanElemenPenilaian()
    {
        return $this->kumpulanElemenPenilaian;
    }
```

```
    public function getFaktorPengali()
    {
        return $this->faktorPengali;
    }


}
```

5.2.4.2 AktivitasId

```
<?php

namespace App\Modules\Skem\Domain\Model\Base;

use Ramsey\Uuid\Uuid;

class AktivitasId
{
    private $id;

    public function __construct($id = null)
    {
        $this->id = $id ? : Uuid::uuid4()->toString();
    }

    public function id()
    {
        return $this->id;
    }

    public function equals(AktivitasId $aktivitasId)
    {
        return $this->id === $aktivitasId->id;
    }

}
```

5.2.4.3 ElemenPenilaian

```
<?php

namespace App\Modules\Skem\Domain\Model\Base;

class ElemenPenilaian
{
    protected $idKegiatanSkem;
    protected $idKriteriaSkem;
    protected $idAspekNilai;
    protected $aspekNilai;
    protected $kriteriaSkem;
    protected $bobot;
    protected $isView;

    public function __construct(
        string $idKriteriaSkem,
        string  $idAspekNilai,
        string $aspekNilai,
        string $kriteriaSkem,
        int $bobot
    )
    {
        $this->idKriteriaSkem = $idKriteriaSkem;
        $this->idAspekNilai = $idAspekNilai;
        $this->aspekNilai = $aspekNilai;
        $this->kriteriaSkem = $kriteriaSkem;
        $this->bobot = (int)$bobot;
    }

    public static function make(
        $idKriteriaSkem,
        $idAspekNilai,
        $aspekNilai,
        $kriteriaSkem,
        $bobot
    )
    {
```

```
        $elemenPenilaian = new ElemenPenilaian(
            $idKriteriaSkem,
            $idAspekNilai,
            $aspekNilai,
            $kriteriaSkem,
            $bobot
        );
        $elemenPenilaian->isView(0);
        return $elemenPenilaian;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        $idKriteriaSkem,
        $idAspekNilai,
        $aspekNilai,
        $kriteriaSkem,
        $bobot
    )
    {
        $elemenPenilaian = new ElemenPenilaian(
            $idKriteriaSkem,
            $idAspekNilai,
            $aspekNilai,
            $kriteriaSkem,
            $bobot
        );
        $elemenPenilaian->isView(1);
        $elemenPenilaian->setIdKegiatanSkem($idKegiatanSkem);
        return $elemenPenilaian;
    }

    public function setIdKegiatanSkem(AktivitasId $idKegiatanSkem)
    {
        $this->idKegiatanSkem = $idKegiatanSkem;
    }
```

```
    public function isView($isView)
    {
        $this->isView = $isView;
    }

    public function getIdKegiatanSkem()
    {
        return $this->idKegiatanSkem;
    }

    public function getIdKriteriaSkem()
    {
        return $this->idKriteriaSkem;
    }

    public function getIdAspekNilai()
    {
        return $this->idAspekNilai;
    }

    public function getBobot()
    {
        return $this->bobot;
    }

    public function getValue()
    {
        return $this->kriteriaSkem;
    }

    public function getNama()
    {
        return $this->aspekNilai;
    }

//          public static function makeAttribute($idKegiatanSkem,
$idKriteriaSkem, $idAspekNilai, $aspekNilai, $kriteriaSkem, $bobot)
//    {
```

```
//          return new ElemenPenilaian($idKegiatanSkem, $idKriteriaSkem,
$idAspekNilai, $aspekNilai, $kriteriaSkem, $bobot);
//    }
}
```

5.2.4.4 Abdimas

```
<?php

namespace App\Modules\Skem\Domain\Model\Abdimas;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Abdimas extends Aktivitas
{

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
```

```
        $aktivitas = new Abdimas(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Abdimas(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
```

```
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        $faktorPengali = (int) $this->faktorPengali;
        if ($faktorPengali == 1 && $faktorPengali > 0 ) {
            $this->nilaiHuruf = "BC";
        } elseif ($faktorPengali <= 10) {
            $this->nilaiHuruf = "B";
        } elseif ($faktorPengali <= 25) {
            $this->nilaiHuruf = "AB";
        } elseif ($faktorPengali > 25) {
            $this->nilaiHuruf = "A";
        } else {
            throw new PadananNilaiHurufNotFoundException();
```

```
        }
//       $this->nilaiHuruf = 'A';
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

//      public static function makeAbdimas($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $components)
//    {
//        $kumpulanElemenPenilaian = [];
//        foreach ($components as $component) {
//            $att = new ElemenPenilaian($idKegiatanSkem,
//                $component['id_kriteria_skem'],
//                $component['id_aspek_nilai'],
//                $component['aspek_nilai'],
```

```
//                $component['nama'],
//                $component['bobot']
//            );
//            array_push($kumpulanElemenPenilaian, $att);
//        }
//        return new Abdimas($idKegiatanSkem,
//            $idSkem,
//            $idBidangSkem,
//            $tglKlaim,
//            $nama,
//            $idPortofolio,
//            $idJenisPortofolio,
//            $idMhs,
//            $kumpulanElemenPenilaian
//        );
//    }

}
```

5.2.4.5 Internasionalisasi

```
<?php

namespace App\Modules\Skem\Domain\Model\Internasionalisasi;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;
```

```
class Internasionalisasi extends Aktivitas
{

    const ID_PESERTA = 1;
    const ID_PANITIA = 2;
    const ID_MODERATOR = 3;
    const ID_PEMBICARA = 4;

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new Internasionalisasi(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }
```

```
public static function makeView(
    AktivitasId $idKegiatanSkem,
    SkemId  $idSkem,
    int $idBidangSkem,
    string $tglKlaim,
    string $namaKegiatan,
    string $idPortofolio,
    string $idJenisPortofolio,
    string $nilaiHuruf,
    string $nilaiAngka,
    string $kreditSkem,
    string $produkSkem
)
{
    $aktivitas = new Internasionalisasi(
        $idKegiatanSkem,
        $idBidangSkem,
        $tglKlaim,
        $namaKegiatan,
        $idPortofolio,
        $idJenisPortofolio
    );
    $aktivitas->isView(1);

    $aktivitas->setNilaiHuruf($nilaiHuruf);
    $aktivitas->setNilaiAngka($nilaiAngka);
    $aktivitas->setKreditSkem($kreditSkem);
    $aktivitas->setProdukSkem($produkSkem);
    $aktivitas->setIdSkem($idSkem);

    return $aktivitas;
}

public function calculateKreditSkem()
{
    $kreditSkem = 1;
```

```
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        $faktorPengali = (int) $this->faktorPengali;
        if ($faktorPengali == self::ID_PESERTA) {
            $this->nilaiHuruf = "B";
        } elseif ($faktorPengali == self::ID_PANITIA) {
            $this->nilaiHuruf = "AB";
        } elseif ($faktorPengali == self::ID_MODERATOR) {
            $this->nilaiHuruf = "AB";
        } elseif ($faktorPengali == self::ID_PEMBICARA) {
            $this->nilaiHuruf = "A";
        } else {
            throw new PadananNilaiHurufNotFoundException();
        }
//        $this->nilaiHuruf = 'A';
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
```

```
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

}
```

5.2.4.6 Kegiatan

```
<?php

namespace App\Modules\Skem\Domain\Model\Kegiatan;

use App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundException;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Kegiatan extends Aktivitas
{

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
```

```
        string $faktorPengali
    )
    {
        $aktivitas = new Kegiatan(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Kegiatan(
```

```
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        $this->nilaiHuruf = 'A';
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
```

```
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

//      public static function makeKegiatan($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $components)
//    {
//        $kumpulanElemenPenilaian = [];
//        foreach ($components as $component) {
//            $att = new ElemenPenilaian($idKegiatanSkem,
//                $component['id_kriteria_skem'],
//                $component['id_aspek_nilai'],
//                $component['aspek_nilai'],
//                $component['nama'],
//                $component['bobot']
//            );
//            array_push($kumpulanElemenPenilaian, $att);
//        }
//        return new Kegiatan($idKegiatanSkem,
//            $idSkem,
//            $idBidangSkem,
//            $tglKlaim,
```

```
//          $nama,
//          $idPortofolio,
//          $idJenisPortofolio,
//          $idMhs,
//          $kumpulanElemenPenilaian
//      );
//  }


}
```

5.2.4.7 Kompetisi

```
<?php

namespace App\Modules\Skem\Domain\Model\Kompetisi;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;

use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Kompetisi extends Aktivitas
{
    /*
     * Faktor pengali => capaian aktivitas
     */
    const JUARA_SATU = 1;
    const JUARA_DUA = 2;
```

```
    const JUARA_TIGA = 3;
    const JUARA_HARAPAN = 4;
    const FINALIS = 5;
    const PESERTA = 6;
    const PESERTA_DENGAN_PENGAKUAN = 7;

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        int $faktorPengali
    )
    {
        $aktivitas = new Kompetisi(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
```

```
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Kompetisi(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
```

```
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        if ($this->faktorPengali == self::PESERTA) {
            $this->nilaiHuruf = 'D';
        }                elseif                ($this->faktorPengali                ==
self::PESERTA_DENGAN_PENGAKUAN) {
            $this->nilaiHuruf = 'D';
        } elseif ($this->faktorPengali == self::FINALIS) {
            $this->nilaiHuruf = 'C';
        } elseif ($this->faktorPengali == self::JUARA_HARAPAN) {
            $this->nilaiHuruf = 'BC';
        } elseif ($this->faktorPengali == self::JUARA_TIGA) {
            $this->nilaiHuruf = 'B';
        }  elseif ($this->faktorPengali == self::JUARA_DUA) {
            $this->nilaiHuruf = 'AB';
        } elseif ($this->faktorPengali == self::JUARA_SATU) {
            $this->nilaiHuruf = 'A';
        } else {
            throw new PadananNilaiHurufNotFoundException();
        }
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
```

```
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }
}
```

5.2.4.8 LkmmPemandu

```
<?php

namespace App\Modules\Skem\Domain\Model\LkmmPemandu;

use App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundException;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class LkmmPemandu extends Aktivitas
{
    /*
     * Faktor pengali => jenis kegiatan
     */

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
```

```
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new LkmmPemandu(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
```

```
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new LkmmPemandu(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        $this->nilaiHuruf = 'A';
    }
```

```
    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

//        public static function makeLkmmPemandu($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $faktorPengali)
//    {
//        $kumpulanElemenPenilaian = null;
//
//        return new LkmmPemandu($idKegiatanSkem,
//            $idSkem,
//            $idBidangSkem,
//            $tglKlaim,
//            $nama,
//            $idPortofolio,
//            $idJenisPortofolio,
//            $idMhs,
```

```
//          $kumpulanElemenPenilaian,
//          $faktorPengali
//      );
//  }

}
```

5.2.4.9 Magang

```
<?php

namespace App\Modules\Skem\Domain\Model\Magang;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Magang extends Aktivitas
{
    /*
     * Faktor pengali => jenis sertifikat magang
     */
    const ID_MAGANG_BERSERTIFIKAT_KOMPETENSI = 1;
    const ID_MAGANG_BERSERTIFIKAT_INDUSTRI = 2;
    const ID_MAGANG_BERSERTIFIKAT_NON_INDUSTRI = 3;

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
```

```
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new Magang(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
```

```
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Magang(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }


    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        if ($this->faktorPengali ==
self::ID_MAGANG_BERSERTIFIKAT_NON_INDUSTRI) {
```

```
            $this->nilaiHuruf = 'B';
        } elseif ($this->faktorPengali ==
self::ID_MAGANG_BERSERTIFIKAT_INDUSTRI) {
            $this->nilaiHuruf = 'AB';
        } elseif ($this->faktorPengali ==
self::ID_MAGANG_BERSERTIFIKAT_KOMPETENSI) {
            $this->nilaiHuruf = 'A';
        } else {
            throw new PadananNilaiHurufNotFoundException();
        }
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this-
>getNilaiAngka();
    }

//      public static function makeMagang($idKegiatanSkem, $idSkem,
$idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio,
$idMhs, $components, $faktorPengali)
//    {
//        $kumpulanElemenPenilaian = [];
//        foreach ($components as $component) {
//            $att = new ElemenPenilaian($idKegiatanSkem,
//                $component['id_kriteria_skem'],
```

```
//                $component['id_aspek_nilai'],
//                $component['aspek_nilai'],
//                $component['nama'],
//                $component['bobot']
//            );
//            array_push($kumpulanElemenPenilaian, $att);
//        }
//        return new Magang($idKegiatanSkem,
//            $idSkem,
//            $idBidangSkem,
//            $tglKlaim,
//            $nama,
//            $idPortofolio,
//            $idJenisPortofolio,
//            $idMhs,
//            $kumpulanElemenPenilaian,
//            $faktorPengali
//        );
//    }

}
```

5.2.4.10 Mahasiswa

```
<?php

namespace App\Modules\Skem\Domain\Model\Mahasiswa;

use App\Modules\Skem\Domain\Model\Semester\Semester;
use App\Modules\Skem\Domain\Model\Skem\Skem;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use Mockery\Exception;


class Mahasiswa
{
    private $idMhs;
```

```
    private $nama;
    private $thnAngkatan;
    private $nrp;
    private $kumpulanSemester;

    public function __construct(MahasiswaId $idMhs, string $nama, int $thnAngkatan, string $nrp)
    {
        $this->idMhs = $idMhs;
        $this->nama = $nama;
        $this->thnAngkatan = (int)$thnAngkatan;
        $this->nrp = $nrp;
    }

    /*
     * Getter Function Group
     */

    public function getIdMhs()
    {
        return $this->idMhs;
    }

    public function getNama()
    {
        return $this->nama;
    }

    public function getThnAngkatan()
    {
        return $this->thnAngkatan;
    }

    public function getNrp()
    {
        return $this->nrp;
    }

    public function getSemester(SemesterId $idSmt)
```

```
{
   foreach ($this->kumpulanSemester as $semester)
   {
      if ($semester->getIdSmt()->id() == $idSmt->id()) {
        $result = $semester;
        return $result;
      }
   }
   return null;
}

public function getSemuaSemester()
{
   return $this->kumpulanSemester;
}

/*
 * Setter Function Group
 */

public function addSemester(Semester $semester)
{
   if ($this->getIdMhs()->id() == $semester->getIdMhs()->id()) {
      $this->kumpulanSemester[] = $semester;
   }
   else {
      throw new Exception();
   }
}

public function addSkem(Skem $skem)
{
   foreach ($this->getSemuaSemester() as $semester) {
      if ($skem->getIdSmt()->id() == $semester->getIdSmt()->id()) {
            $semester->setSkem($skem);
            return 0;
      }
```

```
        }
    }

    public function addKumpulanSkem(array $kumpulanSkem)
    {
        foreach($kumpulanSkem as $skem) {
            $this->addSkem($skem);
        }
    }
}
```

5.2.4.11 Ormawa

```
<?php

namespace App\Modules\Skem\Domain\Model\Ormawa;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Ormawa extends Aktivitas
{
    /*
     * Faktor pengali => jabatan
     */

    const ID_PENGURUS_INTI = 1;
    const ID_MENTERI = 2;
    const ID_ANGGOTA_PENGURUS = 3;
    const ID_MAGANG = 4;
```

```
    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new Ormawa(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
```

```
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Ormawa(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
```

```
    {
        if ($this->faktorPengali == self::ID_MAGANG) {
            $this->nilaiHuruf = 'BC';
        }                elseif                ($this->faktorPengali                ==
self::ID_ANGGOTA_PENGURUS) {
            $this->nilaiHuruf = 'B';
        } elseif ($this->faktorPengali == self::ID_MENTERI) {
            $this->nilaiHuruf = 'AB';
        } elseif ($this->faktorPengali == self::ID_PENGURUS_INTI) {
            $this->nilaiHuruf = 'A';
        } else {
            throw new PadananNilaiHurufNotFoundException();
        }
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
```

```
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

//      public static function makeOrmawa($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $components, $faktorPengali)
//  {
//      $kumpulanElemenPenilaian = [];
//      foreach ($components as $component) {
//          $att = new ElemenPenilaian($idKegiatanSkem,
//              $component['id_kriteria_skem'],
//              $component['id_aspek_nilai'],
//              $component['aspek_nilai'],
//              $component['nama'],
//              $component['bobot']
//          );
//          array_push($kumpulanElemenPenilaian, $att);
//      }
//      return new Ormawa($idKegiatanSkem,
//          $idSkem,
//          $idBidangSkem,
//          $tglKlaim,
//          $nama,
//          $idPortofolio,
//          $idJenisPortofolio,
//          $idMhs,
//          $kumpulanElemenPenilaian,
//          $faktorPengali
//      );
//  }

}
```

5.2.4.12 Pelatihan

```
<?php

namespace App\Modules\Skem\Domain\Model\Pelatihan;
```

```
use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Pelatihan extends Aktivitas
{
    /*
     * Faktor pengali => jenis pelatihan
     */

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new Pelatihan(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
```

```
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Pelatihan(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
```

```
    $aktivitas->setIdSkem($idSkem);

    return $aktivitas;
}

public function calculateKreditSkem()
{
    $kreditSkem = 1;
    foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
        $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
    }
    $this->kreditSkem = $kreditSkem;
}

public function calculateNilaiHuruf()
{
    $this->nilaiHuruf = 'A';
}

public function calculateNilaiAngka()
{
    if ($this->nilaiHuruf == 'D') {
        $this->nilaiAngka = 1.5;
    } elseif ($this->nilaiHuruf =='C') {
        $this->nilaiAngka = 2.0;
    } elseif ($this->nilaiHuruf == 'BC') {
        $this->nilaiAngka = 2.5;
    } elseif ($this->nilaiHuruf == 'B') {
        $this->nilaiAngka = 3.0;
    } elseif ($this->nilaiHuruf == 'AB') {
        $this->nilaiAngka = 3.5;
    } elseif ($this->nilaiHuruf == 'A') {
        $this->nilaiAngka = 4.0;
    } else {
        throw new PadananNilaiAngkaNotFoundException();
    }
}
```

```
    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

//      public static function makePelatihan($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $faktorPengali)
//    {
//        $kumpulanElemenPenilaian = null;
//
//        return new Pelatihan($idKegiatanSkem,
//            $idSkem,
//            $idBidangSkem,
//            $tglKlaim,
//            $nama,
//            $idPortofolio,
//            $idJenisPortofolio,
//            $idMhs,
//            $kumpulanElemenPenilaian,
//            $faktorPengali
//        );
//    }

}
```

5.2.4.13 PertukaranMhs

```
<?php

namespace App\Modules\Skem\Domain\Model\PertukaranMhs;

use App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundException;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
```

```
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class PertukaranMhs extends Aktivitas
{

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new PertukaranMhs(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
```

```
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new PertukaranMhs(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);

        return $aktivitas;
    }

    public function calculateKreditSkem()
    {
        $kreditSkem = 1;
        foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
```

```
            $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
        }
        $this->kreditSkem = $kreditSkem;
    }

    public function calculateNilaiHuruf()
    {
        $this->nilaiHuruf = 'B'; // karena posisi otomatis peserta
    }

    public function calculateNilaiAngka()
    {
        if ($this->nilaiHuruf == 'D') {
            $this->nilaiAngka = 1.5;
        } elseif ($this->nilaiHuruf =='C') {
            $this->nilaiAngka = 2.0;
        } elseif ($this->nilaiHuruf == 'BC') {
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this-
>getNilaiAngka();
    }
```

```
//         public static function makePertukaranMhs($idKegiatanSkem, $idSkem, $idBidangSkem, $tglKlaim, $nama, $idPortofolio, $idJenisPortofolio, $idMhs, $components)
//     {
//         $kumpulanElemenPenilaian = [];
//         foreach ($components as $component) {
//             $att = new ElemenPenilaian($idKegiatanSkem,
//                 $component['id_kriteria_skem'],
//                 $component['id_aspek_nilai'],
//                 $component['aspek_nilai'],
//                 $component['nama'],
//                 $component['bobot']
//             );
//             array_push($kumpulanElemenPenilaian, $att);
//         }
//         return new PertukaranMhs($idKegiatanSkem,
//             $idSkem,
//             $idBidangSkem,
//             $tglKlaim,
//             $nama,
//             $idPortofolio,
//             $idJenisPortofolio,
//             $idMhs,
//             $kumpulanElemenPenilaian
//         );
//     }

}
```

5.2.4.14 Semester

```
<?php

namespace App\Modules\Skem\Domain\Model\Semester;

use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Skem\Skem;
```

```
class Semester
{
    private $idMhs;
    private $idSmt;
    private $semesterKe;
    private $nama;
    private $skem;


    public function __construct(MahasiswaId $idMhs, SemesterId $idSmt,
int $semesterKe, string $nama)
    {
        $this->idMhs = $idMhs;
        $this->idSmt = $idSmt;
        $this->semesterKe = $semesterKe;
        $this->nama = $nama;

    }

    /*
     * Setter Function Group
     */

    public function setSkem(Skem $skem)
    {
        $this->skem = $skem;
    }

    /*
     * Getter Function Group
     */

    public function getSkem()
    {
        return $this->skem;
    }
```

```
    public function getIdMhs()
    {
        return $this->idMhs;
    }

    public function getIdSmt()
    {
        return $this->idSmt;
    }

    public function getSemesterKe()
    {
        return $this->semesterKe;
    }

    public function getNama()
    {
        return $this->nama;
    }
}
```

5.2.4.15 Skem

```
<?php

namespace App\Modules\Skem\Domain\Model\Skem;

use App\Modules\Skem\Domain\Exception\AktivitasIsNullException;
use
App\Modules\Skem\Domain\Exception\IdBidangSkemNotFoundExcepti
on;
use App\Modules\Skem\Domain\Exception\TipeDataIsFalseException;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Kompetisi\Kompetisi;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Semester\SemesterId;
use App\Modules\Skem\Domain\Model\Wirausaha\Wirausaha;
use App\Modules\Skem\Domain\Model\Magang\Magang;
use App\Modules\Skem\Domain\Model\Ormawa\Ormawa;
```

```
use App\Modules\Skem\Domain\Model\Kegiatan\Kegiatan;
use
App\Modules\Skem\Domain\Model\LkmmPemandu\LkmmPemandu;
use App\Modules\Skem\Domain\Model\Pelatihan\Pelatihan;
use App\Modules\Skem\Domain\Model\Abdimas\Abdimas;
use
App\Modules\Skem\Domain\Model\Internasionalisasi\Internasionalisasi;
use App\Modules\Skem\Domain\Model\PertukaranMhs\PertukaranMhs;

class Skem
{
    const IP_MINIMUM = 2.1;
    const KATEGORI_MINIMUM = 2;
    const STATUS_LULUS_LULUS = 'L';
    const STATUS_LULUS_TIDAK_LULUS = 'T';

    private $kumpulanAktivitas;

    private $idSkem;
    private $tglPenilaian;

    private $idMhs;
    private $idSmt;
    private $semesterKe;
    private $komentar;

    private $statusLulus;
    private $statusAjuan;

    private $totalProdukA;
    private $totalProdukB;
    private $totalProdukC;
    private $totalProdukD;

    private $totalKreditA;
    private $totalKreditB;
    private $totalKreditC;
```

```
    private $totalKreditD;

    private $totalKreditSkem;
    private $totalProdukSkem;
    private $ipSkem;

    private $totalAktivitasA;
    private $totalAktivitasB;
    private $totalAktivitasC;
    private $totalAktivitasD;
    private $totalAktivitasSkem;

    private $kategoriCukup;
    private $ipSkemCukup;
    private $terkunci;

    private $isView;

    public function getIdSkem()
    {
       return $this->idSkem;
    }

    public function getTglPenilaian()
    {
       return $this->tglPenilaian;
    }

    public function getIdMhs()
    {
       return $this->idMhs;
    }

    public function getIdSmt()
    {
       return $this->idSmt;
    }

    public function getSemesterKe()
```

```
    {
        return $this->semesterKe;
    }

    public function getKomentar()
    {
        return $this->komentar;
    }

    public function getStatusLulus()
    {
        return $this->statusLulus;
    }

    public function getStatusAjuan()
    {
        return $this->statusAjuan;
    }

    public function getTotalProdukA()
    {
        return $this->totalProdukA;
    }

    public function getTotalProdukB()
    {
        return $this->totalProdukB;
    }

    public function getTotalProdukC()
    {
        return $this->totalProdukC;
    }

    public function getTotalProdukD()
    {
        return $this->totalProdukD;
```

```
    }

    public function getTotalProdukSkem()
    {
        return $this->totalProdukSkem;
    }

    public function getIpSkem()
    {
        return $this->ipSkem;
    }

    public function getTotalAktivitasSkem()
    {
        return $this->totalAktivitasSkem;
    }

    public function getKumpulanAktivitas()
    {
        return $this->kumpulanAktivitas;
    }

    public function isView(int $isView)
    {
        $this->isView = $isView;
    }

    public function setKomentar($komentar)
    {
        $this->komentar = $komentar;
    }

    public function setKumpulanAktivitas($kumpulanAktivitas)
    {
        $this->kumpulanAktivitas = $kumpulanAktivitas;
    }

    public function setTotalKreditA(int $totalKreditA)
    {
```

```
    $this->totalKreditA = $totalKreditA;
}

public function setTotalKreditB(int $totalKreditB)
{
    $this->totalKreditB = $totalKreditB;
}

public function setTotalKreditC(int $totalKreditC)
{
    $this->totalKreditC = $totalKreditC;
}

public function setTotalKreditD(int $totalKreditD)
{
    $this->totalKreditD = $totalKreditD;
}

public function setTotalKreditSkem(int $totalKreditSkem)
{
    $this->totalKreditSkem = $totalKreditSkem;
}

public function setTotalProdukA(int $totalProdukA)
{
    $this->totalProdukA = $totalProdukA;
}

public function setTotalProdukB(int $totalProdukB)
{
    $this->totalProdukB = $totalProdukB;
}

public function setTotalProdukC(int $totalProdukC)
{
    $this->totalProdukC = $totalProdukC;
}
```

```
public function setTotalProdukD(int $totalProdukD)
{
   $this->totalProdukD = $totalProdukD;
}

public function setTotalProdukSkem(int $totalProdukSkem)
{
   $this->totalProdukSkem = $totalProdukSkem;
}

public function setIpSkem($ipSkem)
{
   $this->ipSkem = $ipSkem;
}

public function setTotalAktivitasA(int $totalAktivitasA)
{
   $this->totalAktivitasA = $totalAktivitasA;
}

public function setTotalAktivitasB(int $totalAktivitasB)
{
   $this->totalAktivitasB = $totalAktivitasB;
}

public function setTotalAktivitasC(int $totalAktivitasC)
{
   $this->totalAktivitasC = $totalAktivitasC;
}

public function setTotalAktivitasD(int $totalAktivitasD)
{
   $this->totalAktivitasD = $totalAktivitasD;
}

public function setTotalAktivitasSkem(int $totalAktivitasSkem)
{
   $this->totalAktivitasSkem = $totalAktivitasSkem;
```

```
}

public function setKategoriCukup($kategoriCukup)
{
   $this->kategoriCukup = $kategoriCukup;
}

public function calculateKategoriCukup()
{
   $flag = 0;
   if ($this->totalAktivitasA > 0) {
      $flag += 1;
   }
   if ($this->totalAktivitasB > 0) {
      $flag += 1;
   }
   if ($this->totalAktivitasC > 0) {
      $flag += 1;
   }
   if ($this->totalAktivitasD > 0) {
      $flag += 1;
   }
   if ($flag >= self::KATEGORI_MINIMUM) {
      $this->kategoriCukup = 1;
   } else {
      $this->kategoriCukup = 0;
   }
}

public function setIpSkemCukup($ipSkemCukup)
{
   $this->ipSkemCukup = $ipSkemCukup;
}

public function calculateIpSkemCukup()
{
   $minimumIp = self::IP_MINIMUM;
```

```
        if ($this->ipSkem >= $minimumIp) {
            $this->ipSkemCukup = 1;
        } else {
            $this->ipSkemCukup = 0;
        }
    }

    public function setStatusLulus($statusLulus)
    {
        $this->statusLulus = $statusLulus;
    }

    public function calculateStatusLulus()
    {
        if ($this->ipSkemCukup == 1 && $this->kategoriCukup == 1) {
            $this->statusLulus = self::STATUS_LULUS_LULUS;
        } else {
            $this->statusLulus = self::STATUS_LULUS_TIDAK_LULUS;
        }
    }

    public function setStatusAjuan($statusAjuan)
    {
        $this->statusAjuan = $statusAjuan;
    }

    public function addTotalAktivitasA()
    {
        $this->totalAktivitasA += 1;
    }

    public function addTotalAktivitasB()
    {
        $this->totalAktivitasB += 1;
    }

    public function addTotalAktivitasC()
    {
        $this->totalAktivitasC += 1;
```

```
    }

    public function addTotalAktivitasD()
    {
        $this->totalAktivitasD += 1;
    }

    public function calculateTotalAktivitasSkem()
    {
        $this->totalAktivitasSkem = $this->totalAktivitasA + $this-
>totalAktivitasB + $this->totalAktivitasC + $this->totalAktivitasD;
    }

    public function addTotalKreditA(int $kreditSkem)
    {
        $this->totalKreditA += $kreditSkem;
    }

    public function addTotalKreditB(int $kreditSkem)
    {
        $this->totalKreditB += $kreditSkem;
    }

    public function addTotalKreditC(int $kreditSkem)
    {
        $this->totalKreditC += $kreditSkem;
    }

    public function addTotalKreditD(int $kreditSkem)
    {
        $this->totalKreditD += $kreditSkem;
    }

    public function calculateTotalKreditSkem()
    {
        $this->totalKreditSkem = $this->totalKreditA + $this->totalKreditB
+ $this->totalKreditC + $this->totalKreditD;
```

```
    }

    public function addTotalProdukA(int $produkSkem)
    {
        $this->totalProdukA += $produkSkem;
    }

    public function addTotalProdukB(int $produkSkem)
    {
        $this->totalProdukB += $produkSkem;
    }

    public function addTotalProdukC(int $produkSkem)
    {
        $this->totalProdukC += $produkSkem;
    }

    public function addTotalProdukD(int $produkSkem)
    {
        $this->totalProdukD += $produkSkem;
    }

    public function calculateTotalprodukSkem()
    {
        $this->totalProdukSkem = $this->totalProdukA + $this-
>totalProdukB + $this->totalProdukC + $this->totalProdukD;
    }

    public function calculateIpSkem()
    {
        if ($this->totalKreditSkem == 0) {
            $this->ipSkem = 0;
        } else {
            $this->ipSkem = $this->totalProdukSkem / $this-
>totalKreditSkem;
        }
    }

    public function resetSkem()
```

```
    {
        $this->totalKreditA = 0;
        $this->totalKreditB = 0;
        $this->totalKreditC = 0;
        $this->totalKreditD = 0;
        $this->totalKreditSkem = 0;

        $this->totalProdukA = 0;
        $this->totalProdukB = 0;
        $this->totalProdukC = 0;
        $this->totalProdukD = 0;
        $this->totalProdukSkem = 0;

        $this->ipSkem = 0;

        $this->totalAktivitasA = 0;
        $this->totalAktivitasB = 0;
        $this->totalAktivitasC = 0;
        $this->totalAktivitasD = 0;
        $this->totalAktivitasSkem = 0;

        $this->kategoriCukup = 0;
        $this->ipSkemCukup = 0;

        $this->statusAjuan = 0;
        $this->statusLulus = self::STATUS_LULUS_TIDAK_LULUS;
        $this->terkunci = 0;

        $this->kumpulanAktivitas = [];
    }

    public function addAktivitas($aktivitas)
    {
        if ($aktivitas == null) {
            throw new AktivitasIsNullException("aktivitas di proses
pembuatan model skem");
        }
```

```
        if ((is_subclass_of($aktivitas, 'App\Modules\Skem\Domain\Model\Base\Aktivitas') || (get_class($aktivitas) == 'App\Modules\Skem\Domain\Model\Base\Aktivitas')) == false) {
            throw new TipeDataIsFalseException(get_class($aktivitas), 'Aktivitas');
        }

        if ($this->isView == 1) {
            return 0; // throw exception
        }
        $aktivitas->setIdSkem($this->getIdSkem());
        array_push($this->kumpulanAktivitas, $aktivitas);
        switch ($aktivitas->getIdBidangSkem()) {
            case 2: // Kompetisi
            case 3: // Wirausaha
            case 4: // Magang
                $this->addTotalAktivitasA();
                $this->addTotalKreditA($aktivitas->getKreditSkem());
                $this->addTotalProdukA($aktivitas->getProdukSkem());
                break;
            case 6: // Ormawa
            case 7: // Kegiatan dan LkmmPemandu
            case 10: // pelatihan
                $this->addTotalAktivitasB();
                $this->addTotalKreditB($aktivitas->getKreditSkem());
                $this->addTotalProdukB($aktivitas->getProdukSkem());
                break;
            case 8: // Abdimas
                $this->addTotalAktivitasC();
                $this->addTotalKreditC($aktivitas->getKreditSkem());
                $this->addTotalProdukC($aktivitas->getProdukSkem());
                break;
            case 11: // Internasionalisasi dan Pertukaran Mhs
                $this->addTotalAktivitasD();
                $this->addTotalKreditD($aktivitas->getKreditSkem());
                $this->addTotalProdukD($aktivitas->getProdukSkem());
                break;
```

```
            default:
                throw new IdBidangSkemNotFoundException($aktivitas->getIdBidangSkem());
        }

        $this->calculateTotalAktivitasSkem();
        $this->calculateTotalKreditSkem();
        $this->calculateTotalprodukSkem();
        $this->calculateIpSkem();
        $this->calculateIpSkemCukup();
        $this->calculateKategoriCukup();
        $this->calculateStatusLulus();
    }

    public function addKumpulanAktivitas(array $kumpulanAktivitas)
    {
        foreach ($kumpulanAktivitas as $aktivitas) {
            $this->addAktivitas($aktivitas);
        }
    }

    /**
     * Skem constructor.
     * @param SkemId $idSkem
     * @param MahasiswaId $idMhs
     * @param SemesterId $idSmt
     * @param int $semesterKe
     */
    public function __construct(
        SkemId $idSkem,
        MahasiswaId $idMhs,
        SemesterId $idSmt,
        int $semesterKe
    )
    {
        $this->idSkem = $idSkem;
        $this->idMhs = $idMhs;
```

```
        $this->idSmt = $idSmt;
        $this->semesterKe = $semesterKe;
        $this->kumpulanAktivitas = [];
    }

    public static function make(
        SkemId $idSkem,
        MahasiswaId $idMhs,
        SemesterId $idSmt,
        int $semesterKe
    )
    {
        $skem = new Skem($idSkem, $idMhs, $idSmt, $semesterKe);

        $skem->setTotalKreditA(0);
        $skem->setTotalKreditB(0);
        $skem->setTotalKreditC(0);
        $skem->setTotalKreditD(0);
        $skem->setTotalKreditSkem(0);

        $skem->setTotalProdukA(0);
        $skem->setTotalProdukB(0);
        $skem->setTotalProdukC(0);
        $skem->setTotalProdukD(0);
        $skem->setTotalProdukSkem(0);

        $skem->setIpSkem(0);

        $skem->setTotalAktivitasA(0);
        $skem->setTotalAktivitasB(0);
        $skem->setTotalAktivitasC(0);
        $skem->setTotalAktivitasD(0);
        $skem->setTotalAktivitasSkem(0);

        $skem->setKategoriCukup(0);
        $skem->setIpSkemCukup(0);

        $skem->setStatusAjuan(0);
        $skem->setStatusLulus(self::STATUS_LULUS_TIDAK_LULUS);
```

```
        $skem->setKomentar(null);
        $skem->isView(0);

        return $skem;
    }

    public static function makeView(
        SkemId $idSkem,
        MahasiswaId $idMhs,
        SemesterId $idSmt,
        int $semesterKe,
        string $statusAjuan,
        string $statusLulus,
        int $totalProdukA,
        int $totalProdukB,
        int $totalProdukC,
        int $totalProdukD,
        int $totalProdukSkem,
        float $ipSkem,
        $komentar
    )
    {
        $skem = new Skem($idSkem, $idMhs, $idSmt, $semesterKe);
        $skem->setTotalProdukA($totalProdukA);
        $skem->setTotalProdukB($totalProdukB);
        $skem->setTotalProdukC($totalProdukC);
        $skem->setTotalProdukD($totalProdukD);
        $skem->setTotalProdukSkem($totalProdukSkem);
        $skem->setIpSkem($ipSkem);
        $skem->setStatusAjuan($statusAjuan);
        $skem->setStatusLulus($statusLulus);
        $skem->setKomentar($komentar);
        $skem->isView(1);

        return $skem;
    }
```

```
}
```

5.2.4.16 Validator

```
<?php

namespace App\Modules\Skem\Domain\Model\Validator;

class Validator
{
    private $idSdm;
    private $nama;

    public function __construct(ValidatorId $idSdm, string $nama)
    {
        $this->idSdm = $idSdm;
        $this->nama = $nama;
    }

    public function getIdSdm()
    {
        return $this->idSdm;
    }

    public function getNama()
    {
        return $this->nama;
    }
}
```

5.2.4.17 Wirausaha

```
<?php

namespace App\Modules\Skem\Domain\Model\Wirausaha;

use
App\Modules\Skem\Domain\Exception\PadananNilaiAngkaNotFoundEx
ception;
```

```
use
App\Modules\Skem\Domain\Exception\PadananNilaiHurufNotFoundEx
ception;
use App\Modules\Skem\Domain\Model\Base\AktivitasId;
use App\Modules\Skem\Domain\Model\Base\ElemenPenilaian;
use App\Modules\Skem\Domain\Model\Base\Aktivitas;
use App\Modules\Skem\Domain\Model\Mahasiswa\MahasiswaId;
use App\Modules\Skem\Domain\Model\Skem\SkemId;

class Wirausaha extends  Aktivitas
{
    /*
     * Faktor pengali => jumlah pendapatan
     */

    public static function make(
        AktivitasId $idKegiatanSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        array $kumpulanElemenPenilaian,
        string $faktorPengali
    )
    {
        $aktivitas = new Wirausaha(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas-
>setKumpulanElemenPenilaian($kumpulanElemenPenilaian);
        $aktivitas->setFaktorPengali($faktorPengali);
```

```
        $aktivitas->isView(0);

        $aktivitas->calculateNilaiHuruf();
        $aktivitas->calculateNilaiAngka();
        $aktivitas->calculateKreditSkem();
        $aktivitas->calculateProdukSkem();

        return $aktivitas;
    }

    public static function makeView(
        AktivitasId $idKegiatanSkem,
        SkemId  $idSkem,
        int $idBidangSkem,
        string $tglKlaim,
        string $namaKegiatan,
        string $idPortofolio,
        string $idJenisPortofolio,
        string $nilaiHuruf,
        string $nilaiAngka,
        string $kreditSkem,
        string $produkSkem
    )
    {
        $aktivitas = new Wirausaha(
            $idKegiatanSkem,
            $idBidangSkem,
            $tglKlaim,
            $namaKegiatan,
            $idPortofolio,
            $idJenisPortofolio
        );
        $aktivitas->isView(1);

        $aktivitas->setNilaiHuruf($nilaiHuruf);
        $aktivitas->setNilaiAngka($nilaiAngka);
        $aktivitas->setKreditSkem($kreditSkem);
        $aktivitas->setProdukSkem($produkSkem);
        $aktivitas->setIdSkem($idSkem);
```

```
    return $aktivitas;
}

public function calculateKreditSkem()
{
    $kreditSkem = 1;
    foreach ($this->kumpulanElemenPenilaian as $elemenPenilaian) {
        $kreditSkem = $kreditSkem * $elemenPenilaian->getBobot();
    }
    $this->kreditSkem = $kreditSkem;
}

public function calculateNilaiHuruf()
{
    $jumlahPendapatan = (int)$this->faktorPengali;

    if ($jumlahPendapatan < 20000000) {
        $this->nilaiHuruf = 'C';
    } elseif ($jumlahPendapatan < 50000000) {
        $this->nilaiHuruf = 'BC';
    } elseif ($jumlahPendapatan < 100000000) {
        $this->nilaiHuruf = 'B';
    } elseif ($jumlahPendapatan <= 500000000) {
        $this->nilaiHuruf = 'AB';
    } elseif ($jumlahPendapatan > 500000000) {
        $this->nilaiHuruf = 'A';
    } else {
        throw new PadananNilaiHurufNotFoundException();
    }
}

public function calculateNilaiAngka()
{
    if ($this->nilaiHuruf =='C') {
        $this->nilaiAngka = 2.0;
    } elseif ($this->nilaiHuruf == 'BC') {
```

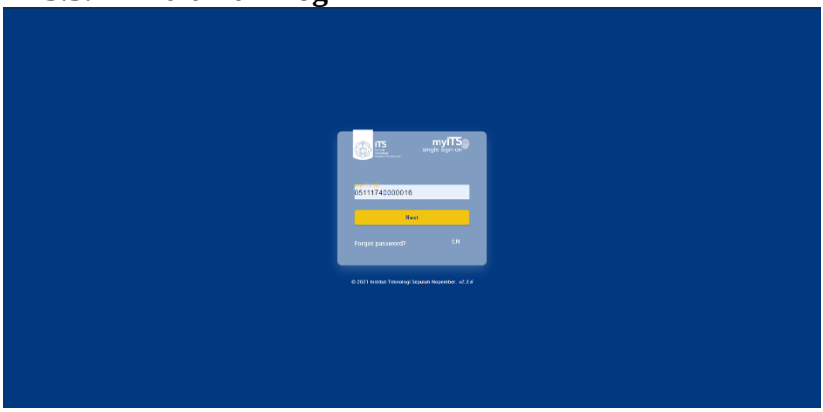
```
            $this->nilaiAngka = 2.5;
        } elseif ($this->nilaiHuruf == 'B') {
            $this->nilaiAngka = 3.0;
        } elseif ($this->nilaiHuruf == 'AB') {
            $this->nilaiAngka = 3.5;
        } elseif ($this->nilaiHuruf == 'A') {
            $this->nilaiAngka = 4.0;
        } else {
            throw new PadananNilaiAngkaNotFoundException();
        }
    }

    public function calculateProdukSkem()
    {
        $this->produkSkem = $this->getKreditSkem() * $this->getNilaiAngka();
    }

}
```

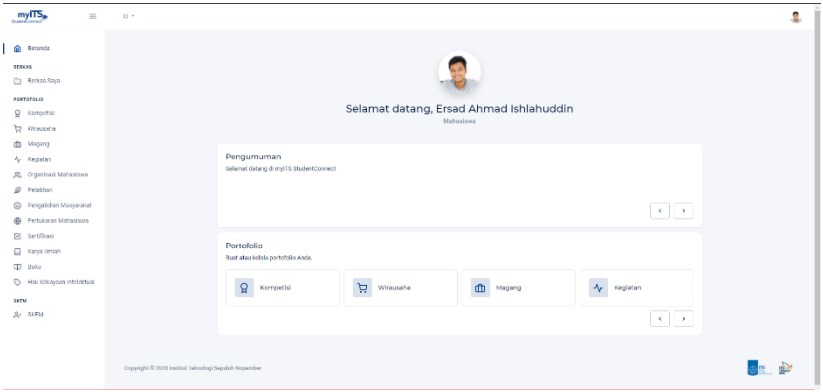
5.3 Implementasi UI

5.3.1 Halaman Login



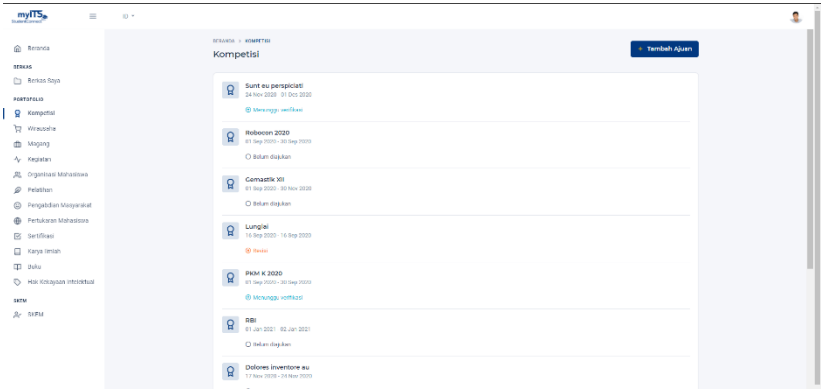
Gambar 2: Halaman Login

5.3.2 Halaman Beranda



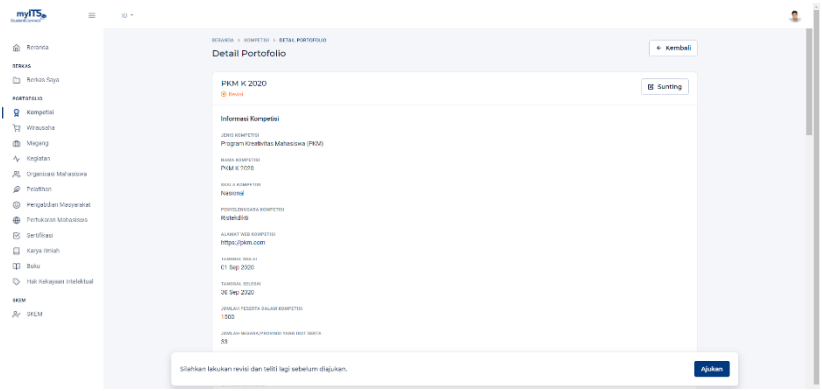
Gambar 3: Halaman Beranda

5.3.3 Halaman Daftar Portofolio



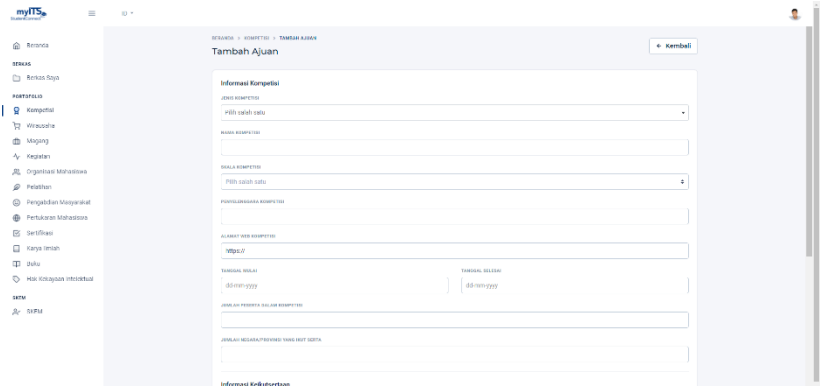
Gambar 4: Halaman Daftar Portofolio

5.3.4 Halaman Detail Portofolio



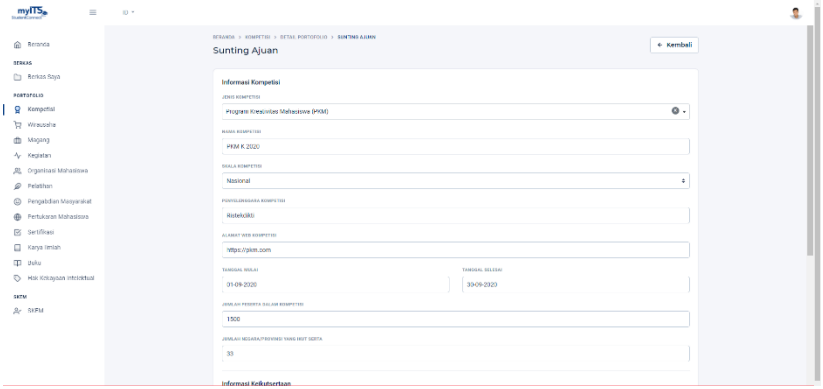
Gambar 5: Halaman Detail Portofolio

5.3.5 Halaman Tambah Portofolio



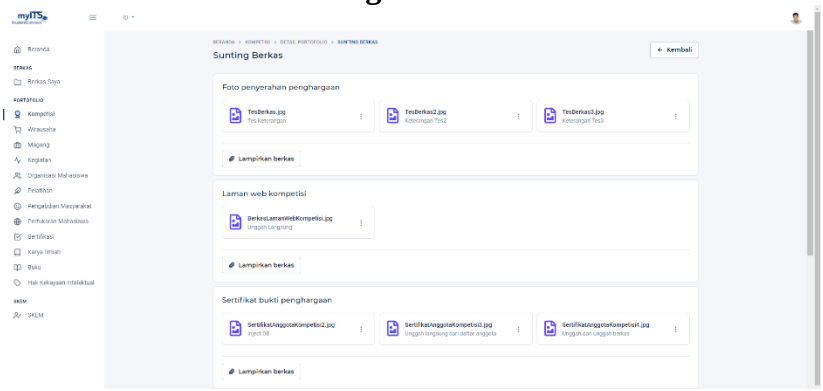
Gambar 6: Halaman Tambah Portofolio

5.3.6 Halaman Sunting Portofolio



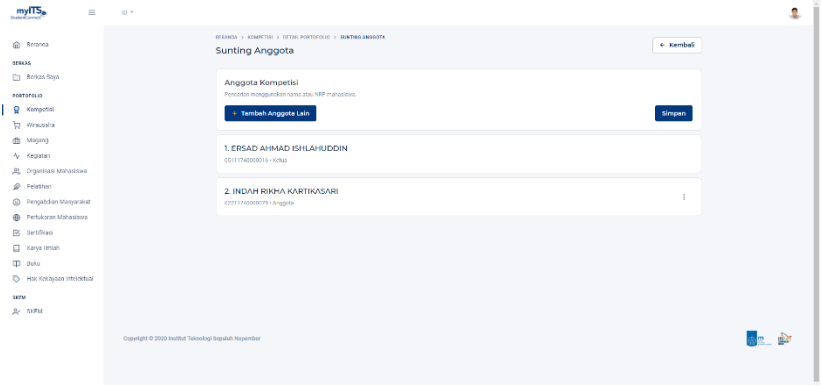
Gambar 7: Halaman Sunting Portofolio

5.3.7 Halaman Sunting Berkas Portofolio



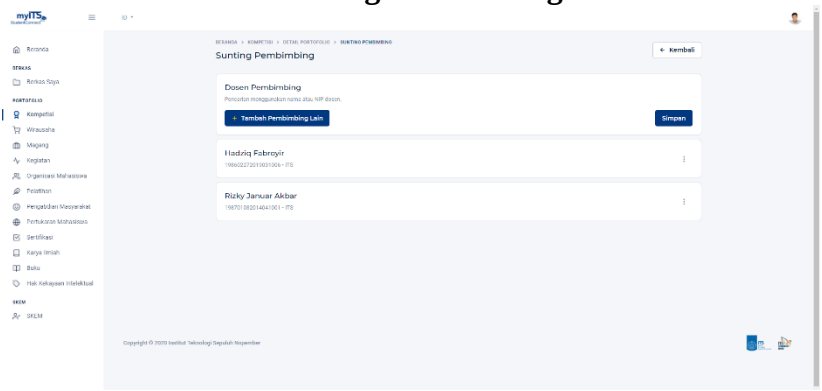
Gambar 8: Halaman Sunting Berkas Portofolio

5.3.8 Halaman Sunting Anggota Portofolio



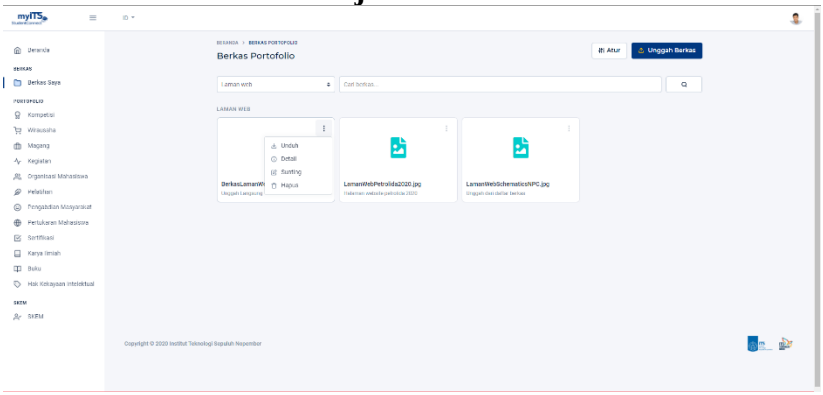
Gambar 9: Halaman Sunting Anggota Portofolio

5.3.9 Halaman Sunting Pembimbing Portofolio



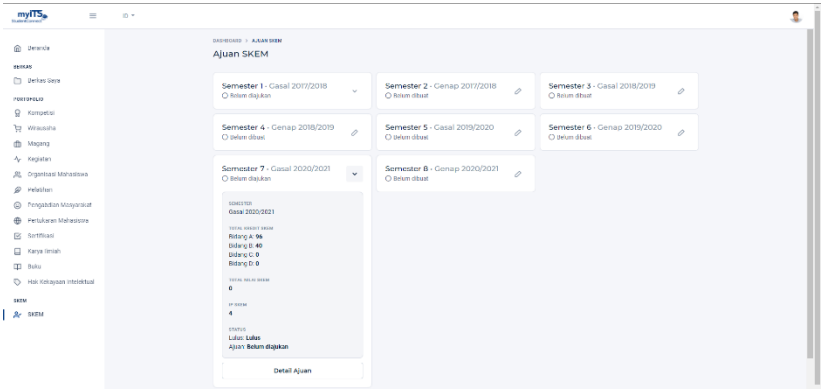
Gambar 10: Halaman Sunting Pembimbing Kompetisi

5.3.10 Halaman Manajemen Berkas



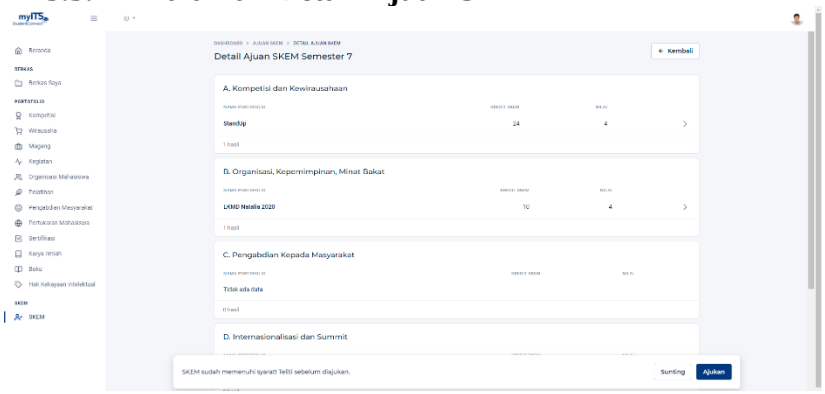
Gambar 11: Halaman Manajemen Berkas

5.3.11 Halaman SKEM



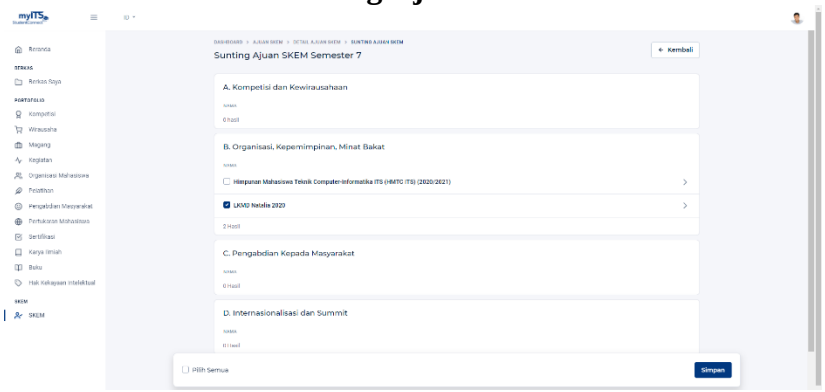
Gambar 12: Halaman SKEM

5.3.12 Halaman Detail Ajuan SKEM



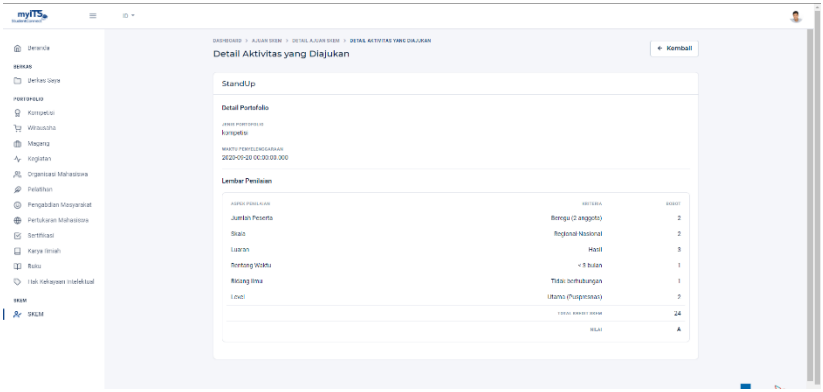
Gambar 13: Halaman Detail Ajuan SKEM

5.3.13 Halaman Sunting Ajuan SKEM



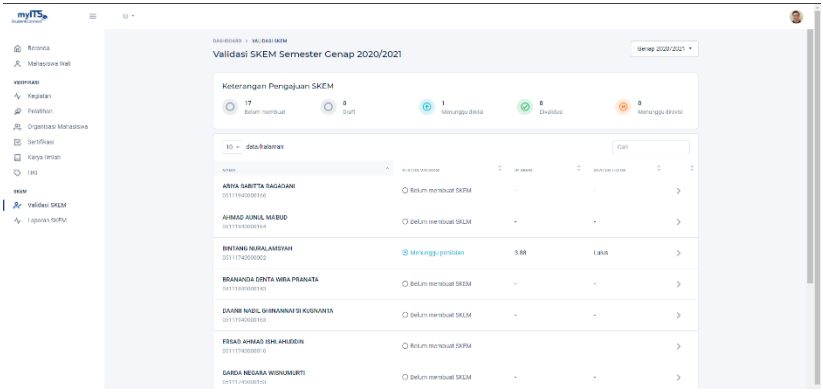
Gambar 14: Halaman Sunting Ajuan SKEM

5.3.14 Halaman Detail Penilaian SKEM



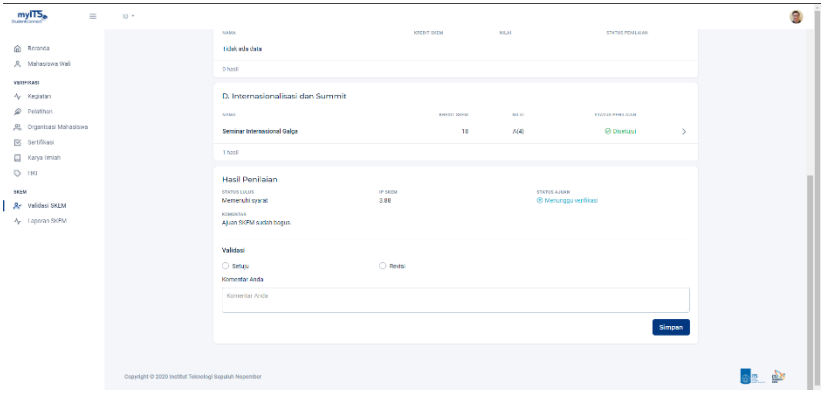
Gambar 15: Halaman Detail Penilaian SKEM

5.3.15 Halaman Validasi SKEM



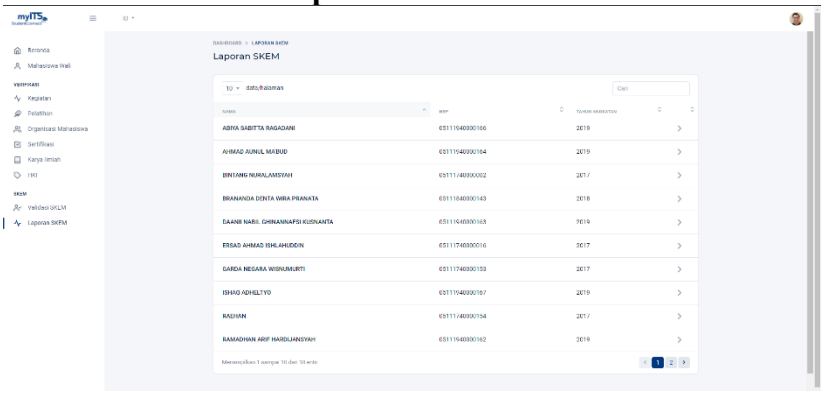
Gambar 16: Halaman Validasi SKEM

5.3.16 Halaman Detail Validasi SKEM



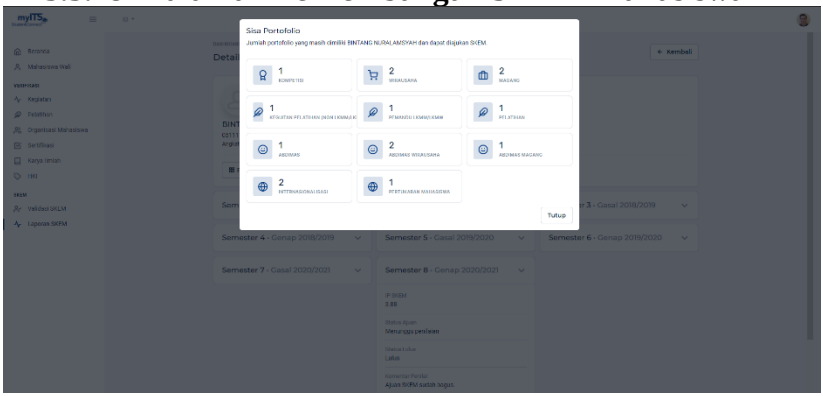
Gambar 17: Halaman Detail Validasi SKEM

5.3.17 Halaman Laporan SKEM



Gambar 18: Halaman Laporan SKEM

5.3.18 Halaman Perkembangan SKEM Mahasiswa



Gambar 19: Halaman Perkembangan SKEM Mahasiswa

[Halaman ini sengaja dikosongkan]

BAB VI
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba terhadap Modul Portofolio dan SKEM myITS StudentConnect. Pengujian dilakukan untuk memastikan kualitas perangkat lunak yang dibangun dan kesesuaian hasil eksekusi perangkat lunak dengan analisis dan perancangan perangkat lunak.

6.1. Skenario Pengujian

6.1.1. Melakukan login

1. Membuka halaman login.
2. Memasukkan username dan password myITS SSO.
3. Menekan tombol sign in.

6.1.2. Menambahkan portofolio

1. Melakukan login.
2. Memilih kategori portofolio pada menu.
3. Menekan tombol tambah ajuan.
4. Mengisi data yang dibutuhkan.
5. Menekan tombol simpan.

6.1.3. Melihat detail portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.

6.1.4. Menyunting portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting.

5. Mengisi data yang dibutuhkan.
6. Menekan tombol simpan.

6.1.5. Menghapus portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting.
5. Menekan tombol hapus portofolio.
6. Menekan tombol hapus.

6.1.6. Mengunggah berkas portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada kelengkapan berkas.
5. Menekan tombol lampirkan berkas.
6. Menekan tombol unggah berkas.
7. Mengisi data yang dibutuhkan.
8. Menekan tombol kirim.

6.1.7. Memilih berkas portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada kelengkapan berkas.
5. Menekan tombol lampirkan berkas.
6. Menekan tombol pilih dari berkas saya.
7. Memilih berkas yang dibutuhkan.
8. Menekan tombol kirim.

6.1.8. Menambah anggota pada portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada anggota.

5. Menekan tombol tambah anggota lain.
6. Mencari anggota.
7. Menekan tombol simpan.

6.1.9. Menghapus anggota pada portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada anggota.
5. Menekan tanda titik tiga di anggota.
6. Menekan tombol hapus.
7. Menekan tombol hapus.

6.1.10. Menambah pembimbing portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada pembimbing.
5. Menekan tombol tambah pembimbing lain.
6. Mencari pembimbing.
7. Menekan tombol simpan.

6.1.11. Menghapus pembimbing portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.
3. Membuka salah satu portofolio.
4. Menekan tombol sunting pada pembimbing.
5. Menekan tanda titik tiga di pembimbing.
6. Menekan tombol hapus.
7. Menekan tombol hapus.

6.1.12. Mengajukan portofolio

1. Melakukan login.
2. Membuka halaman daftar portofolio.

3. Membuka salah satu portofolio.
4. Menekan tombol ajukan.

6.1.13. Menambah berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tombol unggah berkas.
4. Mengisi data yang dibutuhkan.
5. Menekan tombol kirim.

6.1.14. Menyunting berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tanda titik tiga pada berkas.
4. Menekan tombol sunting.
5. Mengisi data yang ingin diubah.
6. Menekan tombol kirim.

6.1.15. Menghapus berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tanda titik tiga pada berkas.
4. Menekan tombol hapus.
5. Menekan tombol kirim.

6.1.16. Mengunduh berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tanda titik tiga pada berkas.
4. Menekan tombol unduh.

6.1.17. Melihat detail berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tanda titik tiga pada berkas.
4. Menekan tombol detail.

6.1.18. Mencari berkas portofolio

1. Melakukan login.
2. Membuka menu berkas saya.
3. Menekan tombol atur.
4. Mengisi filter sesuai kebutuhan.
5. Menekan tanda cari.

6.1.19. Membuat ajuan SKEM

1. Melakukan login.
2. Membuka menu skem.
3. Menekan tanda pensil pada semester tertentu.
4. Memilih portofolio.
5. Menekan tombol simpan dan lanjutkan.

6.1.20. Menyunting ajuan SKEM

1. Melakukan login.
2. Membuka menu skem.
3. Menekan tanda panah bawah pada semester tertentu.
4. Menekan tombol detail ajuan.
5. Menekan tombol sunting.
6. Memilih portofolio.
7. Menekan tombol simpan dan lanjutkan.

6.1.21. Mengajukan SKEM

1. Melakukan login.
2. Membuka menu skem.
3. Menekan tanda panah bawah pada semester tertentu.
4. Menekan tombol detail ajuan.
5. Menekan tombol ajukan.
6. Menekan tombol ajukan.

6.1.22. Melihat detail ajuan SKEM

1. Melakukan login.
2. Membuka menu skem.

3. Menekan tanda panah bawah pada semester tertentu.
4. Menekan tombol detail ajuan.

6.1.23. Melihat detail portofolio pada ajuan SKEM

1. Melakukan login.
2. Membuka menu skem.
3. Menekan tanda panah bawah pada semester tertentu.
4. Menekan tombol detail ajuan.
5. Menekan salah satu portofolio.

6.1.24. Memvalidasi SKEM

1. Melakukan login.
2. Memilih peran sebagai dosen.
3. Memilih menu validasi SKEM.
4. Memilih ajuan untuk divalidasi.
5. Mengisi form validasi.
6. Menekan tombol simpan.

6.1.25. Melihat perkembangan SKEM anak wali

1. Melakukan login.
2. Memilih peran sebagai dosen.
3. Memilih menu laporan SKEM.
4. Memilih salah satu mahasiswa.
5. Menekan semester tertentu.
6. Menekan portofolio tersisa.

6.2. Evaluasi Pengujian

Pada subbab ini akan diberikan hasil evaluasi dari pengujian-pengujian yang telah dilakukan. Hasil evaluasi pengujian dapat dilihat pada Tabel 27.

Tabel 27: Hasil evaluasi pengujian aplikasi sesuai kebutuhan

Nomor	Deskripsi Kebutuhan	Status
UC-001	Melakukan login mahasiswa	Berhasil
UC-002	Menambahkan portofolio	Berhasil

UC-003	Melihat detail portofolio	Berhasil
UC-004	Menyunting portofolio	Berhasil
UC-005	Menghapus portofolio	Berhasil
UC-006	Mengunggah berkas portofolio	Berhasil
UC-007	Memilih berkas portofolio	Berhasil
UC-008	Menambah anggota pada portofolio	Berhasil
UC-009	Menghapus anggota pada portofolio	Berhasil
UC-010	Menambah pembimbing portofolio	Berhasil
UC-011	Menghapus pembimbing portofolio	Berhasil
UC-012	Mengajukan portofolio	Berhasil
UC-013	Menambah berkas portofolio	Berhasil
UC-014	Menyunting berkas portofolio	Berhasil
UC-015	Menghapus berkas portofolio	Berhasil
UC-016	Mengunduh berkas portofolio	Berhasil
UC-017	Melihat detail berkas portofolio	Berhasil
UC-018	Mencari berkas portofolio	Berhasil
UC-019	Membuat ajuan SKEM	Berhasil
UC-020	Menyunting ajuan SKEM	Berhasil
UC-021	Mengajukan SKEM	Berhasil
UC-022	Melihat detail ajuan SKEM	Berhasil
UC-023	Melihat detail portofolio pada ajuan SKEM	Berhasil
UC-024	Memvalidasi SKEM	Berhasil
UC-025	Melihat perkembangan SKEM anak wali	Berhasil

[Halaman ini sengaja dikosongkan]

BAB VII
KESIMPULAN DAN SARAN

Kesimpulan yang didapat setelah melakukan pembuatan Modul Portofolio dan SKEM myITS StudentConnect adalah sebagai berikut:

a. Aplikasi yang dibangun cukup sesuai dengan permintaan dan dapat dengan mudah dioperasikan oleh pengguna.
b. Dengan adanya Modul Portofolio dan SKEM myITS StudentConnect, mahasiswa ITS dapat memiliki satu aplikasi yang datanya dapat digunakan untuk banyak keperluan, sedangkan dari pihak ITS dapat memanfaatkan data kemahasiswaan dengan lebih baik.

DAFTAR PUSTAKA

[1] Qwords (2020). Sistem Informasi - Qwords [online] Available at: https://qwords.com/blog/pengertian-sistem-informasi-dan-jenisnya [Accessed 30 Desember 2020].
[2] Hostinger (2020). HTML - Hostinger [online] Available at: https://www.hostinger.co.id/tutorial/apa-itu-html [Accessed 30 Desember 2020].
[3] Hostinger (2019). CSS - Hostinger [online] Available at: https://www.hostinger.co.id/tutorial/apa-itu-css [Accessed 30 Desember 2020].
[4] Hostinger (2017). Javascript - Hostinger [online] Available at : https://www.hostinger.co.id/tutorial/apa-itu-javascript [Accessed 30 Desember 2020].
[5] Niagahoster (2020). PHP - Niagahoster [online] Available at: https://www.niagahoster.co.id/blog/pengertian-php [Accessed 30 Desember 2020].
[6] Niagahoster (2019). Laravel Framework - Niagahoster [online] Available at: https://www.niagahoster.co.id/blog/laravel-adalah [Accessed 30 Desember 2020].
[7] Dicoding (2020). Visual Studio Code - Dicoding [online] Available at: https://www.dicoding.com/blog/microsoft-visual-studio-code [Accessed 30 Desember 2020].

BIODATA PENULIS

Ersad Ahmad Ishlahuddin, lahir pada tanggal 9 Juli 1999 di Bojonegoro. Penulis merupakan mahasiswa yang sedang menempuh studi di Departemen Informatika Institut Teknologi Sepuluh Nopember (ITS).

Bintang Nuralamsyah, lahir pada tanggal 7 April 1999 di Bondowoso. Penulis merupakan mahasiswa yang sedang menempuh studi di Departemen Informatika Institut Teknologi Sepuluh Nopember (ITS).